Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2011)
J. Bender, K. Erleben, and E. Galin (Editors)

Towards Symmetry Axis based Markerless Motion Capture

Preprint. The definitive version is available at http://diglib.eg.org/.

Philip Hartmann and Svenja Kahn and Ulrich Bockholt and Arjan Kuijper
Fraunhofer IGD, Darmstadt, Germany

Abstract

A natural interaction with virtual environments is one of the key issues for the usability of Virtual Reality applications. Device-free, intuitive interactions with the virtual world can be achieved by capturing the movements of the user with markerless motion capture. In this work we present a markerless motion capture approach which can be used to estimate the human body pose in real-time with a single depth camera. The presented approach requires neither a 3D shape model of the tracked person nor a training phase in which body shapes are learned a priori. Instead, it analyzes the curvature of the human body to estimate the symmetry axes of the body joints. These symmetry axes are then used to calculate the pose of the tracked human in real-time. The presented approach was evaluated qualitatively with a time-of-flight and a Kinect depth camera. Furthermore, quantitative simulation results show that the proposed approach is promising for depth cameras which can reliably capture the surface curvature (and thus the normals) of a person and which have a resolution of at least 320x240 pixel.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality H.5.2 [Computer Graphics]: User Interfaces—Input devices and strategies I.4.8 [Computer Graphics]: Scene Analysis—Motion, Range Data, Tracking I.2.10 [Artificial Intelligence]: Vision and Scene Understanding—Motion,Video Analysis

1. Introduction

One of the key issues for the usability of Virtual Reality applications is a natural interaction with the virtual environment. Most VR applications require the use of specialized interaction devices, for example a spacemouse or a flystick [Zha98] [WPLP07] [Zim08]. On the other hand, a system which could capture the gestures or body movements of the user with motion capture technology would enable device-free, intuitive interactions with the virtual worlds. Current state-of-the-art motion capture technologies have the drawback that they require the installation, calibration and maintenance of complex and expensive multi-camera systems [CMG*10] [Org11]. Furthermore, most motion capture systems are marker-based [Nat11] [Vic11]. The attachment of markers to the human body and the need to wear a specialized marker suit or a data glove can be uncomfortable and hinder a natural interaction with the virtual environment.

To overcome these limitations, a markerless system which captures human movements in real-time with a minimal hardware setup is required for an intuitive and deviceless interaction with virtual worlds. The first system which fulfills these requirements is Microsoft's Kinect which estimates the human pose from the depth images of a single depth camera [SFC*11]. Whereas the Kinect provides ready-to-use markerless motion capture, it has the drawback that it is a proprietary solution which can only be used with the Kinect hardware and with software licenses from PrimeSense or Microsoft. The motion capture approach used for the Kinect requires a computationally very expensive train-

Figure 1: Free-hand, deviceless VR interaction

© The Eurographics Association 2011.

ing step in which hundreds of thousands poses captured with marker-based motion capture are first acquired to simulate and to analyze artificial depth images on a 1000-core cluster [SFC*11]. A motion capture method which requires such an expensive preprocessing step can only be employed with large financial investment: Microsoft spent hundreds of millions dollars for the development of the Kinect [Van11].

Real-time depth images can be captured with time-of-flight cameras as well [OLB06] [KBKL09]. Whereas structured light cameras like the Kinect estimate the depth by projecting a a pattern onto the scene and by analyzing the distortion of the pattern, time-of-flight cameras emit near-infrared modulated light. The distance is calculated by the time it took the light to return to the camera after it was reflected by the scene. Several depth-image based motion capture approaches have been proposed for these depth cameras, either for single body parts [JPL09] [BW09] [HCCL10], for the upper body and the arms [GKK07] [ZDF08] or for full body tracking [PG08]. For a deviceless interaction with virtual environments, the markerless motion capture method should be able to track the overall movements of the user. Thus it should be possible to track the upper body, the arms and the legs. Approaches which track only the pose of single body parts, for example the legs [JPL09] or the hands [BW09], are not feasible for a deviceless, full-body controlled VR interaction. Another requirement is that the motion capture needs to be real-time capable. Motion capture methods which require more than 1000ms per frame for full body tracking [PG08] or more than 100ms per frame for a partial body tracking (which tracks only the arms or the upper body, but not the legs) [GKK07] [ZDF08] [BW09] cannot be used for real-time VR interaction by full body pose estimation.

So far, apart from the approach used for the Kinect [SFC*11], only two methods have been proposed with which the upper body, the arms and the legs can be tracked in real-time and which could thus be suited for markerless VR interaction [KVD09] [GPKT10]. The method proposed by Knoop et al. uses the Iterative Closest Point algorithm to geometrically align the depth image with an approximated cylindrical 3D model of the person [KVD09]. Due to the fact that the pose is estimated by aligning the surface of the cylindrical 3D model with the shape of the real person, the shape of the cylindrical 3D model should match the real shape of the tracked person as well as possible. This can be achieved by adapting the scale factor of the 3D model and the sizes and radii of the cylinders from which it is composed. However, an adaption of each cylinder radius of the 3D avatar model would be tedious for virtual environments because VR system are often used by persons with different sizes and body shapes. In contrast, symmetry axis based motion capture methods as proposed in this paper have the advantage that the radii of the limbs do not need to be known exactly for the specific tracked person. The approach proposed by Ganapathi et al. [GPKT10] does not require a 3D model of the tracked person. However, similar to the Kinect approach, it uses body part recognition for the pose estimation. Therefore a preprocessing step is necessary in which different possible appearances of the feet, the hands and the head are learned. Furthermore, even with a GPU implementation of this method, only 4-10 frames can be evaluated per second, which is too slow for a smooth real-time VR interaction via motion capture.

In this work we present a markerless motion capture approach which can be used to estimate the human body pose in real-time, thus fulfilling this essential requirement for deviceless interaction with VR environments. The presented motion capture method uses the depth image stream of a single depth camera and requires neither a 3D shape model of the tracked person nor a training phase in which body shapes are learned a priori. Instead, it analyzes the curvature of the human body to estimate the rotational symmetry axes of the body joints. These rotational symmetry axes are used to calculate the pose of the tracked human in real-time. In the remainder of this paper we first describe symmetry axes in section 2 and our new motion capture approach, which uses rotational symmetry axes for real-time pose estimation, in section 3. Evaluation results are presented in section 4 and the paper ends with conclusions in section 5.

2. Symmetry Axes

Skeleton structures, for example the skeleton of the human body, can be represented by symmetry axes. The most commonly used symmetry axis is the medial axis introduced by Blum [Blu67]. In the three-dimensional case, the medial axis of an object is the union of the centers of all maximal spheres which fit inside the object. The medial axis corresponds to a *reflectional* symmetry axis [BSTZ06]. Another symmetry axis is the *rotational* symmetry axis of an object. Figure 2 visualizes the rotational symmetry axis of a cylinder.

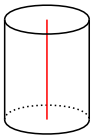


Figure 2: Rotational axis of a cylinder

Most human body parts have a shape which is similar to a cylinder: The arms and legs have a cylindrical shape and the upper body can be approximated with a stretched cylinder. The 3D rotational symmetry axes of the arms, legs and the upper body have the same position as the bones of these body parts and thus human movements can be tracked by calculating the 3D rotational symmetry axes of the body parts. Whereas 2D medial axes have been previously used to estimate the skeleton structure of the projection of human silhouettes to 2D images [BDP*94] [CN08] [YK10], so far no symmetry-axis based method has been proposed for human motion capture in the 3D space.

© The Eurographics Association 2011.

A symmetry axis based motion capture method for VR interaction, which uses the video stream from a single depth camera, needs to fulfill two major requirements: First, it needs to be real-time capable. Second, it needs to compute the symmetry axis from an incomplete point set with large areas of missing data. When a single depth camera is used, the surface of the tracked human is only captured from a single viewpoint and large areas of the body surface are not visible. Most approaches for the calculation of symmetry axes from point clouds can only handle data sets with few missing surface information [OI92] [SLSK07]. However, recently Tagliasacchi et al. [TZCO09] presented a method for the estimation of curve skeletons from imperfect and unordered point clouds with large areas of missing data. This is achieved by using normal information of the 3D points to compensate for missing data (see Figure 3). Premises for this method are that the object is composed from cylindrical regions (except at the joints) and that the point normals are known for the 3D point set. Both preconditions can be met as human body parts have shapes similar to cylinders and as point normals can be calculated for depth images.

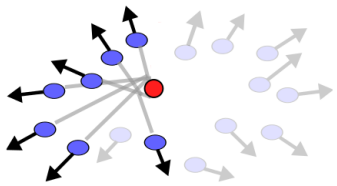


Figure 3: Estimation of the rotational symmetry axis (ROSA): The normals of 3D points can compensate for missing data (figure from [TZCO09], © ACM).

To calculate the rotational symmetry axis (ROSA) of a point set, the method proposed by Tagliasacchi et al. [TZCO09] works on local subsets of the point cloud. To localize the search for a point on the rotational symmetry axis (a ROSA point), first one of the 3D points of the point cloud is selected as an anchor point. Then recursive planar cuts are used to calculate an optimal cutting plane in an iterative manner. An optimal cutting plane is a cutting plane which intersects the anchor point and which is as rotationally symmetric to the normals in its close neighborhood as possible. Given an optimal cutting plane, the ROSA point of this cutting plane is calculated by optimizing the quadratic minimization problem stated in Equation (1) with differentiation. Here r_i^* is the rotational symmetry center, N_i^* are the 3D input points in the local neighbourhood of the cutting plane and $\mathbf{n}(p_j)$ is the normal of point p_j.

$$r_i^* = \underset{x \in \Re^3}{\operatorname{argmin}} \sum_{p_j \in N_i^*} \|(x - p_j) \times \mathbf{n}(p_j)\|^2 \qquad (1)$$

While the method presented by Tagliasacchi et al. fulfills the requirement that the symmetry axis needs to be calculated from an incomplete point cloud [TZCO09], it has two drawbacks which hinder its use for real-time motion capture: First, it is not real-time capable. The skeleton reconstruction of a point cloud with 10.000 points (which would correspond to a depth image with a resolution of 100x100 pixel) takes three minutes with a Matlab implementation. Furthermore, it does not differentiate between the skeletons of different body parts. Rather, it estimates a single skeleton curve for the whole object. Therefore it is not obvious how to map the arbitrary skeleton curve to the different rigid bones of a human skeleton.

3. Motion Capture with Rotational Symmetry Axes

This section describes the new motion capture algorithm which infers the human pose from the depth images of a single depth camera. The presented motion capture algorithm analyzes the shape of the tracked person to estimate the rotational symmetry axis of each tracked skeleton part. It incorporates temporal and spatial knowledge as well as a priori knowledge about the human shape to achieve real-time capability and to map the rotational symmetry axis to the bones of corresponding body parts. Furthermore, our motion capture method builds on the work of Tagliasacchi et al. [TZCO09] by incorporating the estimation of rotational symmetry axis (ROSA) points from incomplete 3D point sets. Our algorithm is based on the following main concepts:

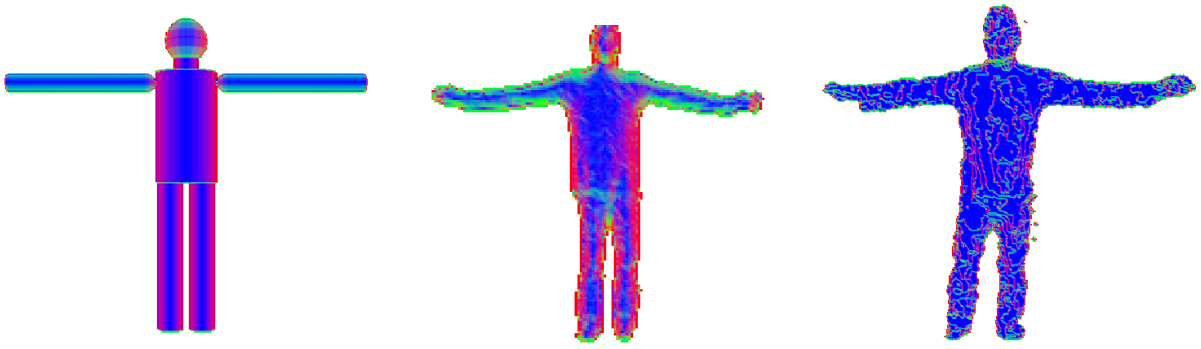
1. We incorporate a priori knowledge about the human skeleton into our algorithm. Therefore, instead of calculating the complete symmetry axis for the whole input point cloud, only a small number of rotational symmetry centers needs to be calculated for each body part.
2. The calculated rotational symmetry axis points are mapped to a human skeleton.
3. Neighbourhood information from the depth image speeds up the point selection for the ROSA point calculations.
4. Significant further speed-up is achieved by incorporating knowledge about the pose of the previous frame into our algorithm.

For the pose estimation, each depth image acquired by the depth camera is first preprocessed (3.1). If the pose was not initialized yet or if it was lost, the pose is (re-)initialized (3.2). If the pose was already tracked in the previous frame, the pose is updated to the current depth image (3.3).

3.1. Depth Image Preprocessing

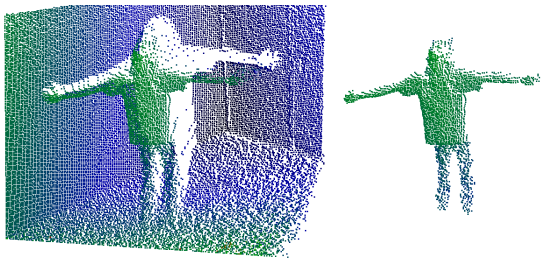
The substeps of the preprocessing are noise reduction, background subtraction, conversion of depth values to euclidean 3D points and point normal estimation. In order to reduce the noise in the depth images, the depth values are smoothed with a bilateral gaussian filter [TM98]. In contrast to non-bilateral filters, bilateral gaussian filters are edge-preserving. This is particularly important for human motion capture

© The Eurographics Association 2011.



(a) Simulated depth image (b) Time-of-flight depth image (c) Kinect depth image

Figure 4: Point normals, visualized as RGB values (x=red, y=green and z = blue)



(a) Input point cloud (b) Without background

Figure 5: Background subtraction

to avoid incorrect smoothing at the jump edges between the person and the background. Then the depth measurements on the surface of the tracked person are identified by background subtraction [WAWB09]. The depth value of each pixel is compared with the mean and the variance of previously recorded background images in order to decide whether the depth difference is significant enough to interpret the depth measurement as a foreground pixel (the background images only need to be acquired once after the installation of the depth camera). Then a connected-component merging is applied to find the largest connected segment. All other depth measurements are discarded.

In the next step, each depth value d_{cam} is converted to a 3D point P_{CCS} in the camera coordinate system (CCS) with Equation (2). Here (p_x, p_y) are the 2D coordinates of the pixel in the pixel coordinate system of the depth image. The focal length (f_x, f_y) and the principal point (c_x, c_y) were estimated with an offline calibration procedure [SBK08]. Figure 5 visualizes the 3D point cloud before and after the filtering and the background subtraction.

The final preprocessing step is the calculation of the point normals. The neighbourhood relations of the 3D points are known from their pixel coordinates in the depth image: The 3D points of four neighboured pixels (forming a square) are divided into two triangles. Then vertex and point normals are calculated for each triangle. Figure 4 visualizes the point normals of different depth images.

$$p_{CCS} = \begin{pmatrix} (p_x - c_x) \cdot \frac{1}{f_x} \cdot d_{cam} \\ (p_y - c_y) \cdot \frac{1}{f_y} \cdot d_{cam} \\ d_{cam} \end{pmatrix} \quad (2)$$

3.2. Pose (Re-)Initialization

Internally, the estimated pose of the tracked person is represented by the joint angles of a human skeleton. An initialization pose helps to adapt the size of the skeleton to the person and to initialize the tracking. The initalization pose corresponds to a "T" pose in which the arms are stretched to the side. For the initialization, the user should face the camera. The size of the virtual skeleton is scaled such that its height and the width of its outstreched arms correspond to the size of the bounding box around the user. If the frame-to-frame tracking gets lost, it can be reinitialized with the initialization pose.

3.3. Frame-to-Frame Tracking

The human pose estimated in the previous depth image is the initial approximation for the pose of the current depth image. To adapt the pose to the new depth image, it is updated in two steps: First, symmetry axes are estimated for each tracked body part. This is accomplished in a fast, real-time capable manner by calculating several rotational symmetry axis points for each body part and by fitting a straight line through the ROSA points. In a second step, the skeleton is aligned with the calculated symmetry axes. In the remainder of this section, both steps are explained in more detail.

© The Eurographics Association 2011.

3.3.1. Calculation of Symmetry Axes

On the bone of each tracked body part (upper body as well as the upper and lower arms and legs), $2 \leq n_{checkpoints} \leq 10$ equally distributed checkpoints are selected. Then a rotational symmetry axis center point is calculated for each checkpoint. This section first describes the conditions which are used to select feasible 3D measurements for the ROSA point calculation. Furthermore, it describes how temporal and spatial relationships of depth image based frame-to-frame tracking are exploited to speed up the selection of 3D input points for the ROSA point calculation such that the presented method gets real-time capable.

Selection of 3D Input Points for ROSA Estimation For each checkpoint c, all neighboured 3D measurements $s \in S$ which fulfill the two conditions stated in Equation (3) and (4) are selected as input points for the ROSA point estimation. First, they should lie in the close neighbourhood of the cutting plane which intersects the checkpoint and which is perpendicular to the bone orientation o calculated in the previous frame (Equation (3)). To select enough 3D measurements for a robust ROSA point estimation, all measurements in a close neighbourhood to this optimal perpendicular cutting plane are selected as input points. δ sets how much the angle between the bone orientation o and the line from the checkpoint c to the 3D measurement s may differ from the perpendicular angle of the optimal cutting plane, which is $\frac{\pi}{2} rad$ respectively $90°$.

$$\frac{\pi}{2} rad - \delta < |\sphericalangle(o,(s-c))| < \frac{\pi}{2} rad + \delta \tag{3}$$

The second condition is stated in Equation (4): To avoid that 3D measurements of other body parts influence the ROSA point calculation, the distance of s to the control point c may not exceed a maximal distance $d_{max} = 1.5r$ to the checkpoint c. Here r is the radius of the tracked body part.

$$|(s-c)| \leq d_{max} \tag{4}$$

Figure 6 visualizes the input points for the ROSA calculation. Five checkpoints were set on each tracked body part. The 3D measurements which are highlighted in cyan fulfill the criterions for a cutting plane and are thus used to calculate the ROSA points. One ROSA point is calculated for each checkpoint by minimizing the quadratic minimization problem stated in Equation (1). Then for each bone of the skeleton the symmetry axis is calculated with a best fit straight line through its ROSA points.

Real-Time Selection of 3D Input Points The key to real-time motion capture with symmetry axes is a fast selection of the 3D input points which are used to calculate a ROSA point on the symmetry axis. In order to achieve the desired real-time capability, our algorithm builds on the fact that depth

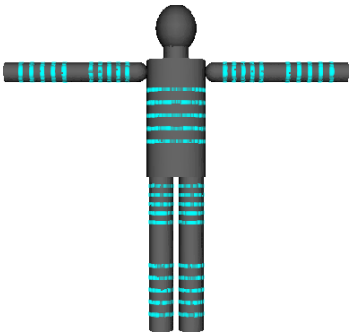


Figure 6: For each checkpoint, all 3D measurements close to the cutting plane which intersects the checkpoint and which is perpendicular to the bone are selected. The selected 3D measurements are highlighted in cyan.

cameras do not acquire unordered 3D point clouds, but structured depth images: For each 2D pixel (i, j) in the pixel coordinate system (PCS) of the depth image the depth camera measures a depth measurement d_{cam}. Each depth measurement can be converted to a 3D point P_{CCS} in the camera coordinate system with Equation (2) (see section 3.1). Therefore neighboured 3D measurements in the camera coordinate system can be found by selecting the 3D measurements of pixels which are neighboured in the 2D pixel coordinate system PCS.

A checkpoint c is not directly linked to 2D coordinates in the pixel coordinate system because it is not a 3D measurement but a 3D point on the symmetry axis calculated from the previous frame. However, the projection of a checkpoint c from the camera coordinate system to the pixel coordinate system (c_{PCS}) can be calculated with Equation (5).

$$c_{PCS} = K \cdot c \tag{5}$$

K is the camera calibration matrix which is composed from the focal length (f_x, f_y), the principal point (c_x, c_y) and the pixel skew, which equals "1" for depth cameras with rectangular pixels.

$$K = \begin{pmatrix} f_x & skew & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{6}$$

After a checkpoint c was projected to the 2D image with Equation (5), the search area S for input points of the ROSA point calculation of this checkpoint is restricted to the 3D measurements whose pixel coordinates are close to c_{PCS} in the 2D image.

© The Eurographics Association 2011.

3.3.2. Skeleton Alignment with Symmetry Axes

The final step of the frame-to-frame tracking is the alignment of the skeleton with the calculated symmetry axes. To account for the hierarchical structure of the human body, the alignment is carried out in a hierarchical manner. First the upper body, then the upper arms and legs and finally the forearms and the lower legs are aligned with the symmetry axes. If the symmetry axis of a body part could not be calculated due to severe occlusions, its symmetry axis remains unchanged until it can be tracked again.

Upper body The first step is the alignment of the upper body of the skeleton with the orientation of its calculated symmetry axis. The position of the upper body on the symmetry axis has one degree of freedom, sliding along the axis. Therefore also its exact position on the symmetry axis needs to be estimated. The position of the upper body on the symmetry axis is unambiguously inferred from the uppermost head position. An orthogonal projection is used to project the uppermost 3D measurement to the closest point on the symmetry axis, p_{head}.

Shoulders For the calculation of the shoulder positions, the upper body is approximated with a cylinder around the symmetry axis whose diameter corresponds to the distance between the shoulders of the human skeleton. Figure 7 and 8 visualize the calculation of the shoulder positions. To assure a robust estimation, the algorithm chooses one of two shoulder position estimation methods. We observed a smooth transition when the algorithm switched from one method to the other. Which method is chosen depends on the angle between the symmetry axis of the upper arm and the symmetry axis of the upper body. If this angle is greater than 45° respectively $\frac{\pi}{4} rad$, the shoulder positions are calculated by intersecting the symmetry axes of the upper arms with the cylinder which approximates the upper body (Figure 7).

For postures where the arms are close to the body, the method which intersects the symmetry axes is not stable: If the arms hang limp, their symmetry axes are approximately parallel to the symmetry axis of the upper body and thus the symmetry axes do not intersect the cylinder at the shoulders. Thus if the angle between the symmetry axis of the arm and the upper body is smaller than 45°, the shoulder positions are estimated by intersecting the symmetry axis of the upper arm with a plane which is perpendicular to the symmetry axis of the upper body and which intersects the position p_{neck} of the neck (see Figure 8). The neck position p_{neck} lies on the symmetry axis of the upper body. Its distance to p_{head} corresponds to the head size.

Hip The intersection point of the symmetry axis of the upper body with the hip (p_{hip}) is the point on the symmetry axis whose distance to p_{head} is the sum of the length of the head and the upper body. Similar to the second method for the estimation of the shoulder positions, the hip positions are calculated by intersecting the symmetry axes of the upper legs with a plane which is perpendicular to $o_{upper\ body}$ and which intersects p_{hip}. Figure 9 visualizes the estimation of the hip position.

Forearms and Lower Legs The final alignment step for each depth image is the alignment of the forearms and the lower legs to the calculated symmetry axes of these body parts. The positions of the elbows are unambiguously defined by the positions of the shoulders and the length and the orientation of the upper arm. Therefore only the orientation of the forearms needs to be set, which is the orientation of their symmetry axes. The same applies to the lower legs.

4. Evaluation

The algorithm was evaluated with regard to its execution time and the accuracy of the calculated pose. For the evaluation of the pose accuracy a simulation was used, providing artificial depth images for specified reference poses. Furthermore, the feasibility of current state-of-the-art depth cameras for symmetry axis based motion capture was evaluated.

4.1. Execution time

The algorithm is implemented in C++. For the evaluation of the processing time a 2.4 Ghz Intel Core 2 Duo processor was used. The execution time of the initialization is 0.74ms for a resolution of 176×144 pixel, 1.97ms for 320×320 pixel, 7.78ms for 640×480 pixel and 30.17ms for 1280×960 pixel. The processing time of the initialization step is approximately linear to the number of depth measurements in the depth image. It is fast enough for real-time processing even for large depth images with 1280×960 depth measurements.

Table 1 gives the number of frames per second which can be calculated with the proposed frame-to-frame tracking. The processing time depends on the number of checkpoints (and thus the number of ROSA points used for the symmetry axis calculation) of each body part. If five checkpoints are used to calculate the symmetry axis of each tracked body part, the presented algorithm is real-time capable with more than 30 frames per second for depth images with up to 640x480 depth measurements. The processing time given by Table 1 is a single core CPU implementation. A further significant speed-up could be achieved easily by parallelizing the ROSA point calculation for each checkpoint.

4.2. Ground Truth Simulation

To evaluate the algorithm with known ground truth data, artificial depth images of an articulated 3D avatar were created. The 3D avatar model is visualized in Figure 10. To create a smooth 3D avatar animation, the joint rotations between the poses of specified key frames were interpolated with spherical linear interpolation. Equation (7) specifies the distance

© The Eurographics Association 2011.

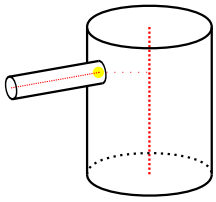


Figure 7: Shoulder estimation (method 1): Intersection of the arm's symmetry axis with a cylinder

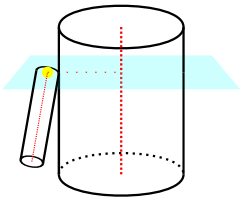


Figure 8: Shoulder estimation (method 2): Intersection of the arm's symmetry axis with a plane

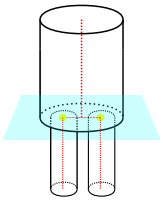
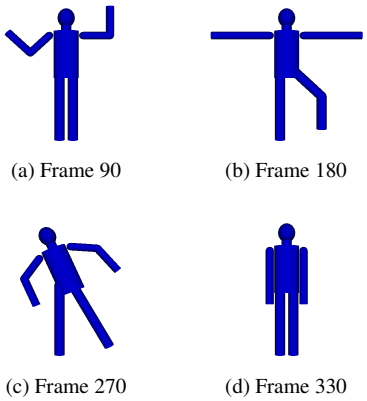


Figure 9: Calculation of hip position: Intersection of the legs' symmetry axes with a plane

Resolution	**Checkpoints**				
	2	**3**	**5**	**7**	**10**
176×144	345	243	163	121	82
320×320	149	106	65	47	34
640×480	73	51	30	21	15
1280×960	13	8	5	3	2

Table 1: Frames per second (frame-to-frame tracking)

metric used for the evaluation of the accuracy of the calculated pose $\tilde{\mathbf{x}}$. The distance metric is the average positional error. n is the number of evaluated body part positions, $p(\mathbf{x_i})$ is the reference 3D position of body part x_i and $p(\tilde{\mathbf{x}}_\mathbf{i})$ is the 3D position which was calculated by the motion capture algorithm. The 3D positions of the shoulders, elbows, hands, hip, knees and feet were compared. For the evaluation of the pose estimation accuracy, five checkpoints were used for each limb. This choice was made because increasing the number of checkpoints per limb up to this value also increased the pose estimation accuracy. In our experiments, the use of more than five checkpoints only caused a very small further enhancement.



(a) Frame 90 (b) Frame 180

(c) Frame 270 (d) Frame 330

Figure 10: Avatar movements of the first sequence

$$D(\mathbf{x},\tilde{\mathbf{x}}) = \frac{1}{n}\sum_{i=1}^{n} \|p(\mathbf{x_i}) - p_i(\tilde{\mathbf{x}}_\mathbf{i})\| \tag{7}$$

The motion capture method was evaluated with two test sequences. In the first sequence the avatar is moved without occlusions. In the second sequence occlusions and challenging movements are simulated. As the accuracy of the pose estimation increased when the number Figure 10 visualizes four avatar poses of the first test sequence. The average positional error of this sequence is plotted in Figure 11. The plot shows that the movements of the whole sequence can be tracked well with depth image resolutions of at least 320×240 depth measurements. The tracking is significantly less accurate for depth images with a resolution of 176×144 measurements. This is due to the fact that only a small number of 3D measurements is available for the calculation of the ROSA points on the arms and the legs if the whole depth image has such a low resolution. To evaluate the effect of noise in the depth images, gaussian noise was added to the depth measurements. Figure 12 visualizes how the accuracy of the pose estimation is influenced by the amount of noise in the depth measurements. σ is the standard deviation of the gaussian noise. Current state-of-the-art depth cameras have a standard deviation of about 1% of the measured distance. A typical interaction setup in which the distance between the camera and the user is 2m thus has a standard deviation of 0.02m. For 3m distance it is 0.03m. In the simulation, the pose estimation seems to be slightly more accurate for light noise (σ = 0.01m) than for noise-free depth images. However, this effect probably only occurs because the 3D avatar model used for the simulation is approximated by planar patches (3D vertex meshes). Adding slight noise to the simulated depth images reduces this planarity and thus seems to result in a slightly more precise estimation of the medial axes. For real depth images (or higher resolution 3D models used for the simulation), the accuracy can be expected to decrease steadily with increasing measurement noise.

Figure 13 visualizes three poses of the second test sequence. In the first two poses the upper body respectively the arms are tilted towards the camera. This makes it difficult to calculate their symmetry axes. In the third pose the

© The Eurographics Association 2011.

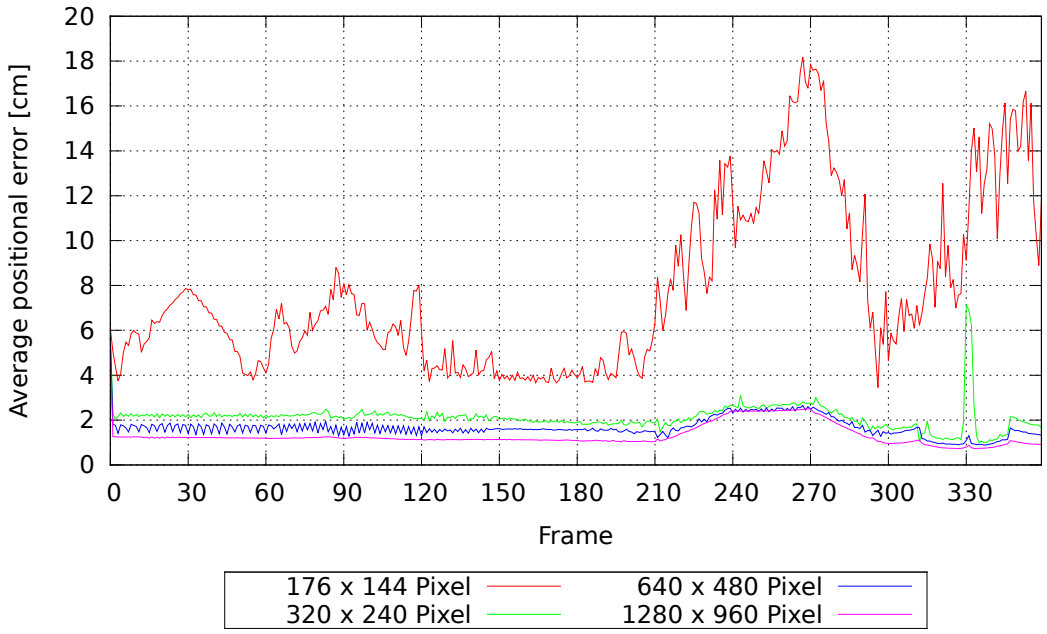


Figure 11: Average positional error of first sequence

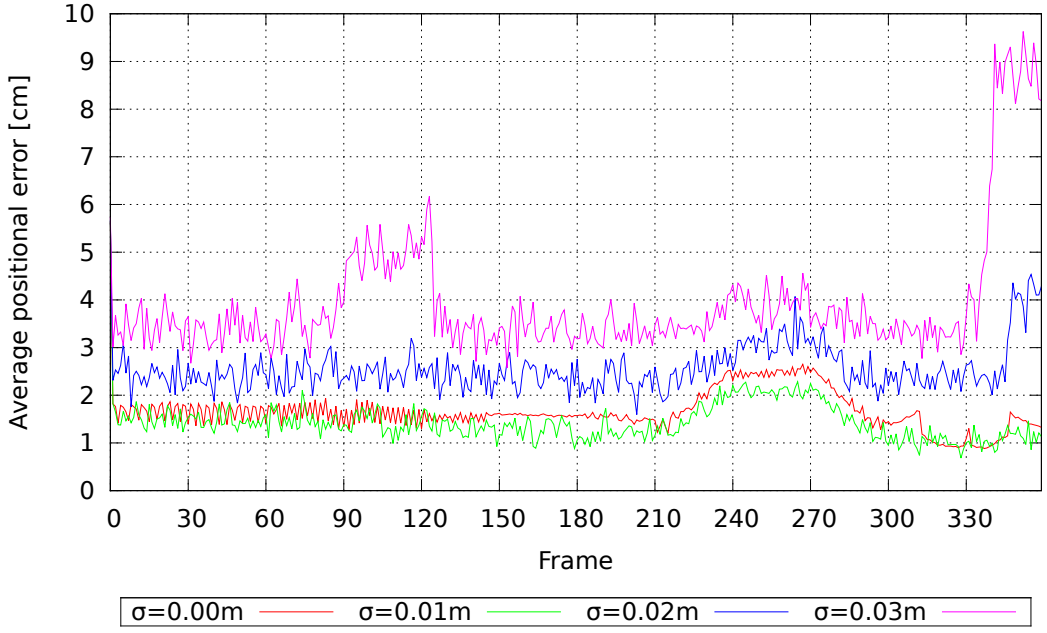
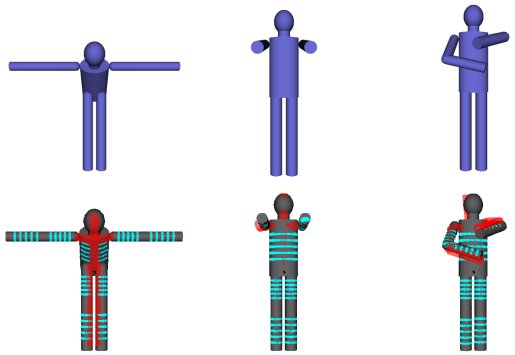


Figure 12: Average positional error of first sequence, noisy depth images (resolution: 640 × 480 pixel)

upper body is partially occluded by the arms. To evaluate these poses independently from each other, the avatar takes up the initialization pose in frame 120 and in frame 180. The average residual for this test sequence is plotted in Figure 14.

For a stable estimation of the poses of this sequence a higher resolution is required than for the first sequence. The accuracy of the estimated pose significantly increases with the resolution of the depth image. The second row of Figure 13

© The Eurographics Association 2011.



(a) Frame 90 (b) Frame 150 (c) Frame 300

Figure 13: First row: Avatar movements of the second sequence. Second row: Calculated pose (1280×640 pixel).

shows the input points for the ROSA point calculation and the estimated pose for depth images with a high resolution of 1280×640 measurements.

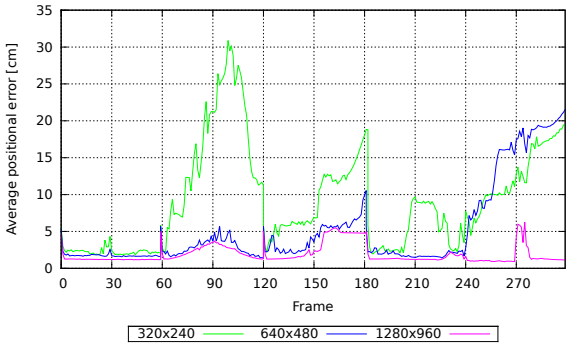


Figure 14: Average positional error of second sequence

4.3. State-of-the-Art Depth Cameras

Depth cameras are subject to fast technological advancements. To evaluate whether the resolution and measurement accuracy of current state-of-the-art depth cameras is already suitable for markerless motion capture via symmetry axes, we recorded test sequences with a SwissRanger 3000 [OLB06] and with a Kinect depth camera. Whereas the SwissRanger 3000 is a time-of-flight camera, the Kinect depth camera uses light coding for the distance measurements. The SwissRanger 3000 measures 176×144 depth measurements and the Kinect depth camera has a resolution of 640×480 pixel. Figure 4 visualizes the point normals of the tracked human. Whereas the surface curvature is well captured with the time-of-flight camera, almost all normals of the Kinect depth image are parallel to the viewing direction of the camera. The surface measured by the Kinect exhibits a pyramidal effect: The measured distance is strongly discretized into few planar layers which are parallel to the image plane of the camera. This effect can be explained by the fact that the Kinect estimates depth values with a depth resolution of only 2^{11} bit. Therefore each depth measurement is strongly discretized. Due to this discretization effect, reliable point normals cannot be calculated and thus the Kinect depth camera is not suited for depth image based symmetry axis calculation. In contrast to the Kinect, the SwissRanger time-of-flight camera measures distances of up to 7.5m with a depth resolution of 2^{14} bit and with an accuracy which is high enough for a reliable normal estimation. However, it has a very low resolution of only 176×144 pixel. Thus the number of measurements on the arms and legs is very small. Even for the noise-free simulated data, this resolution is too small for a robust estimation of the symmetry axes (see Figure 11). We observed the same effect with the depth images captured by the time-of-flight camera. Nevertheless, the time-of-flight camera fulfills the important criterion that reliable normals can be calculated based on its depth images. Thus the future development of time-of-flight cameras with a higher number of depth measurements can be expected to provide suitable depth measurement technology for symmetry axis based motion capture.

5. Conclusion

Real-time depth imaging is subject to rapid technological improvements. Whereas current state-of-the-art depth cameras either have a rather small image resolution or discretize the depth measurements so significantly that normals cannot be inferred from the depth values, the development of depth cameras with a higher resolution and more reliable depth measurements is to be expected in the near future. In this paper we have presented a markerless motion capture algorithm which tracks human movements by estimating the symmetry axes of the human body from the depth images of a single depth camera. Just as real-time depth imaging can enhance the realism of Mixed Reality applications by realistic shadow and occlusion visualization [FKOJ11], depth-image based markerless motion capture can contribute to an intuitive interaction with virtual worlds. The evaluation results of our algorithm show that symmetry axis based motion capture has great potential for depth images with at least 320x240 depth measurements. The presented algorithm aligns body parts with their corresponding symmetry axes. By incorporating knowledge about temporal and spatial neighbourhood relations, the algorithm is able to track human movements in real-time. Thus it fulfills the most important requirement for human motion capture for deviceless interaction with virtual environments. Furthermore, only a single depth camera is required. Therefore the presented algorithm can be used to track the human movements in VR environments without the need to set up complex multi-camera systems. In contrast to the algorithm which is used for Microsoft's Kinect, the presented algorithm is able to

© The Eurographics Association 2011.

track human movements without a learning phase in which hundreds of thousands depth images need to be analyzed.

Acknowledgement

We thank Andrea Tagliasacchi for providing the source code for the ROSA point estimation. This work was partially funded by the German BMBF project Motivotion60+ (16KT0929).

References

[BDP*94] BHARATKUMAR A., DAIGLE K., PANDY M., CAI Q., AGGARWAL J.: Lower limb kinematics of human walking with the medial axis transformation. In *Proceedings of the 1994 IEEE Workshop on Motion of Non-Rigid and Articulated Objects* (1994), pp. 70 –76. 2

[Blu67] BLUM H.: A transformation for extracting new descriptors of shape. In *Models for the Perception of Speech and Visual Form*, Dunn W. W., (Ed.). MIT Press, Cambridge, 1967, pp. 362–380. 2

[BSTZ06] BOUIX S., SIDDIQI K., TANNENBAUM A., ZUCKER S.: Medial axis computation and evolution. In *Statistics and Analysis of Shapes*, Krim H., Yezzi A., (Eds.). Birkhäuser Boston, 2006, pp. 1–28. 2

[BW09] BLEIWEISS A., WERMAN M.: Fusing time-of-flight depth and color for real-time segmentation and tracking. In *Dyn3D '09: Proceedings of the DAGM 2009 Workshop on Dynamic 3D Imaging* (Berlin, Heidelberg, 2009), Springer, pp. 58–69. 2

[CMG*10] CORAZZA S., MÜNDERMANN L., GAMBARETTO E., FERRIGNO G., ANDRIACCHI T.: Markerless motion capture through visual hull, articulated icp and subject specific model generation. *Int. J. Comp. Vision 87* (March 2010), 156–169. 1

[CN08] CHU C.-W., NEVATIA R.: Real-time 3d body pose tracking from multiple 2d images. In *Proceedings of the 5th international conference on Articulated Motion and Deformable Objects (AMDO)* (2008), pp. 42–52. 2

[FKOJ11] FRANKE T., KAHN S., OLBRICH M., JUNG Y.: Enhancing realism of mixed reality applications through real-time depth-imaging devices in x3d. In *Proceedings of the 16th International Conference on 3D Web Technology* (2011), Web3D '11, ACM, pp. 71–79. 9

[GKK07] GREST D., KRÜGER V., KOCH R.: Single view motion tracking by depth and silhouette information. In *Proceedings of the 15th Scandinavian conference on Image analysis* (Berlin, Heidelberg, 2007), SCIA'07, Springer, pp. 719–729. 2

[GPKT10] GANAPATHI V., PLAGEMANN C., KOLLER D., THRUN S.: Real time motion capture using a single time-of-flight camera. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* (2010), pp. 755–762. 2

[HCCL10] HU Z., CHEN M., CHU R., LIM H.: Human arm estimation using convex features in depth images. In *Proceedings of 2010 IEEE 17th International Conference on Image Processing* (2010), pp. 3269 –3272. 2

[JPL09] JENSEN R., PAULSEN R., LARSEN R.: Analyzing gait using a time-of-flight camera. In *Proceedings of the 16th Scandinavian Conference on Image Analysis (SCIA)* (2009), Springer, pp. 21–30. 2

[KBKL09] KOLB A., BARTH E., KOCH R., LARSEN R.: Time-of-flight sensors in computer graphics. In *Proc. Eurographics (State-of-the-Art Report)* (2009), pp. 119–134. 2

[KVD09] KNOOP S., VACEK S., DILLMANN R.: Fusion of 2d and 3d sensor data for articulated body tracking. *Robot. Auton. Syst. 57* (March 2009), 321–329. 2

[Nat11] NATURALPOINT: Optitrack, 2011. http://www.naturalpoint.com/optitrack/. 1

[OI92] OGNIEWICZ R., ILG M.: Voronoi skeletons: Theory and applications. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (1992), pp. 63–69. 3

[OLB06] OGGIER T., LUSTENBERGER F., BLANC N.: Miniature 3d tof camera for real-time imaging. In *PIT* (2006), André E., Dybkjær L., Minker W., Neumann H., Weber M., (Eds.), vol. 4021 of *Lecture Notes in Computer Science*, Springer, pp. 212–216. 2, 9

[Org11] OrganicMotion, 2011. http://www.organicmotion.com. 1

[PG08] PEKELNY Y., GOTSMAN C.: Articulated object reconstruction and markerless motion capture from depth video. *Computer Graphics Forum 27*, 2 (April 2008), 399–408. 2

[SBK08] SCHILLER I., BEDER C., KOCH R.: Calibration of a pmd-camera using a planar calibration pattern together with a multi-camera setup. In *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (2008), vol. XXI. ISPRS Congress, pp. 297–302. 4

[SFC*11] SHOTTON J., FITZGIBBON A., COOK M., SHARP T., FINOCCHIO M., MOORE R., KIPMAN A., BLAKE A.: Real-time human pose recognition in parts from a single depth image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2011), p. 8 pp. 1, 2

[SLSK07] SHARF A., LEWINER T., SHAMIR A., KOBBELT L.: On-the-fly curve-skeleton computation for 3d shapes. *Computer Graphics Forum 26* (2007), 323–328. 3

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision (ICCV)* (1998), pp. 839–846. 3

[TZCO09] TAGLIASACCHI A., ZHANG H., COHEN-OR D.: Curve skeleton extraction from incomplete point cloud. *ACM Transactions on Graphics, (Proceedings SIGGRAPH 2009) 28*, 3 (2009), Article 71, 9 pages. 3

[Van11] VANCE A.: With kinect, microsoft aims for a game changer, 2011. New York Times. 2

[Vic11] Vicon, 2011. http://www.vicon.com/. 1

[WAWB09] WIENTAPPER F., AHRENS K., WUEST H., BOCKHOLT U.: Linear-projection-based classification of human postures in time-of-flight data. In *Proceedings of the 2009 IEEE international conference on Systems, Man and Cybernetics* (2009), SMC'09, IEEE Press, pp. 559–564. 4

[WPLP07] WHITE M., PETRIDIS P., LIAROKAPIS F., PLECINCKX D.: Multimodal mixed reality interfaces for visualizing digital heritage. *International Journal of Architectural Computing 5* (2007), 322–337. 1

[YK10] YOON S. M., KUIJPER A.: Human action recognition using segmented skeletal features. In *20th International Conference on Pattern Recognition - ICPR2010* (2010), IEEE, pp. 3740–3743. 2

[ZDF08] ZHU Y., DARIUSH B., FUJIMURA K.: Controlled human pose estimation from depth image streams. *Computer Vision and Pattern Recognition Workshop 0* (2008), 1–8. 2

[Zha98] ZHAI S.: User performance in relation to 3d input device design. *SIGGRAPH Comput. Graph. 32* (1998), 50–54. 1

[Zim08] ZIMMERMANN P.: Virtual reality aided design. a survey of the use of vr in automotive industry. In *Product Engineering*, Talaba D., Amditis A., (Eds.). Springer Netherlands, 2008, pp. 277–296. 1

© The Eurographics Association 2011.