Real-time Body Tracking with One Depth Camera and Inertial Sensors

Thomas Helten* Meinard Müller† Hans-Peter Seidel* Christian Theobalt*
*Saarland University and MPI Informatik †International Audio Laboratories Erlangen
{thelten,theobalt}@mpi-inf.mpg.de meinard.mueller@audiolabs-erlangen.de

Abstract

In recent years, the availability of inexpensive depth cameras, such as the Microsoft Kinect, has boosted the research in monocular full body skeletal pose tracking. Unfortunately, existing trackers often fail to capture poses where a single camera provides insufficient data, such as non-frontal poses, and all other poses with body part occlusions. In this paper, we present a novel sensor fusion approach for real-time full body tracking that succeeds in such difficult situations. It takes inspiration from previous tracking solutions, and combines a generative tracker and a discriminative tracker retrieving closest poses in a database. In contrast to previous work, both trackers employ data from a low number of inexpensive body-worn inertial sensors. These sensors provide reliable and complementary information when the monocular depth information alone is not sufficient. We also contribute by new algorithmic solutions to best fuse depth and inertial data in both trackers. One is a new visibility model to determine global body pose, occlusions and usable depth correspondences and to decide what data modality to use for discriminative tracking. We also contribute with a new inertial-based pose retrieval, and an adapted late fusion step to calculate the final body pose.

1. Introduction

In recent years, the advent of new and inexpensive cameras that measure 2.5D depth images has triggered extensive research in monocular human pose tracking. Most of the trackers introduced so far can be classified into three families—discriminative approaches and generative approaches, and approaches combining both strategies. While discriminative trackers detect cues in the depth image and derive a pose hypothesis from them using a retrieval strategy, generative trackers optimize for the parameters of a human model to best explain the observed depth image. Combining discriminative and generative approaches, hybrid trackers have shown good results for fast motions in real-time scenarios where tracked actors face the camera more or less frontally. However, noise in the depth data, and the ambiguous representation of human poses in depth images are still a challenge and often lead to trackig errors, even if all body parts are actually exposed to the camera. In addition, if large parts of the body are occluded from view, tracking of the full pose is not possible. Using multiple depth cameras can partially remedy the problem [19], but does not eradicate occlusion problems, and is not always practical in home user scenarios. Depth data alone may thus not be sufficient to capture poses accurately in such challenging scenarios. In this paper, we show that fusing a depth tracker with an additional sensor modality, which provides information complementary to the 2.5D depth video, can overcome these limitations. In particular, we use the orientation data obtained from a sparse set of inexpensive inertial measurement devices fixed to the arms, legs, the trunk and the head of the tracked person. We include this additional information as stabilizing evidence in a hybrid tracker that combines generative and discriminative pose computation. Our approach enables us to track fast and dynamic motions, including non-frontal poses and poses with significant self-occlusions, accurately and in real-time.

Contributions. Our method is the first to adaptively fuse inertial and depth information in a combined generative and discriminative monocular pose estimation framework. To enable this, we contribute with a novel visibility model for determining which parts of the body are visible to the depth camera. This model tells what data modality is reliable and can be used to infer the pose, and enables us to more robustly infer global body orientation even in challenging poses, see Sect. 4. Our second contribution is a generative tracker that fuses optical and inertial cues depending on body part visibility, and finds pose parameters via optimization, see Sect. 5. As a third contribution, we introduce two separate retrieval schemes for handling optical and inertial cues for retrieving database poses during discriminative tracking, see Sect. 6. The final pose is found in a late fusion step which uses the results of both trackers mentioned

*This work was funded by the ERC Starting Grant "CapReal".

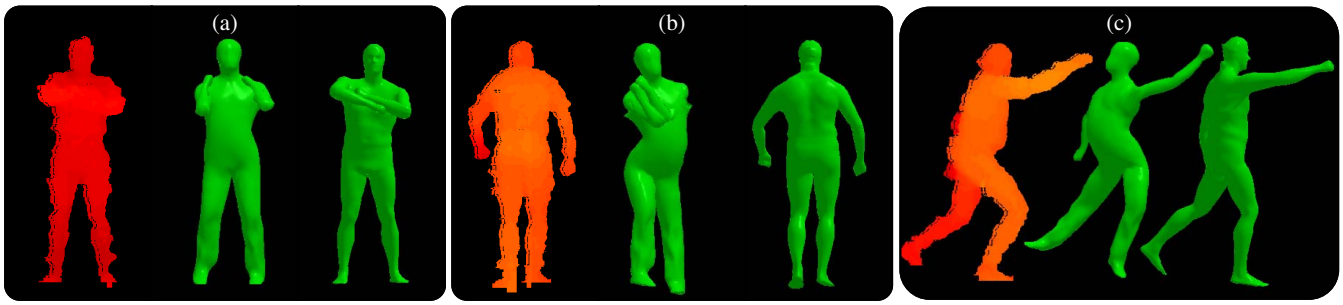

Figure 1. Three typical failure cases of a current real-time tracker combining generative and discriminative pose estimation [1] (left: input depth image; middle: recovered pose of body model with catastrophic pose errors; right: significantly better result using our approach): **(a)** Occluded body parts, **(b)** non-frontal poses, **(b)** and both at the same time.

above, see Sect. 7. We evaluate our proposed tracker on an extensive dataset including calibrated depth images, inertial sensor data, as well as ground-truth data obtained with a traditional marker-based mocap system, see Sect. 8. This dataset is publicly available[1]. We also show qualitatively and quantitatively that it accurately captures poses even under stark occlusion where other trackers fail.

2. Related Work

Marker-less pose estimation from multi-view video has been a long-standing problem in computer vision, and nowadays mature solutions exist, see [11] for an overview. Recently, so-called depth cameras that measure 2.5D geometry information in real-time have emerged [6, 21]. Many monocular tracking algorithms use this depth data for human pose estimation. They can be classified in discriminative approaches, generative approaches and hybrid approaches, reviewed in the following. A discriminative strategy based on body part detectors that also estimated body part orientations on depth images was presented in [9]. Body part detectors and a mapping to a kinematic skeleton are used in [22] to track full-body poses at interactive frame rates. The approach [13] uses regression forests based on depth features to estimate the joint positions of the tracked person without the need for a kinematic model of its skeleton. Later [4], further increased the accuracy, by being able to also detect some occluded joints in non-frontal poses. Finally, also using depth features and regression forests, [16] generate correspondences between body parts and a pose and size parametrized human model that is optimized in real-time using a one-shot optimization approach. While showing good results on single frame basis, these approaches cannot deduce true poses of body parts that are invisible in the camera.

By using kinematic body models with simple shape primitives, the pose of an actor can be found using a generative strategy. The body model is fitted to depth data or to a combination of depth and image features [5, 8]. [2] propose a generative depth-based tracker using a modified energy function that incorporates empty space information, as well as inter-penetration constraints. An approach that uses multiple depth cameras for pose estimation which reduces the occlusion problem is presented in [19]. The approach is not real-time capable, though. With all these depth-based methods, real-time pose estimation is still a challenge, tracking may drift, and with exception to [19] the employed shape models are rather coarse which impairs pose estimation accuracy.

Salzmann *et al.* [12] combine generative and discriminative approaches, with the goal to reconstruct general 3D deformable surfaces. Soon after that, [3] showed a hybrid approach specialized to reconstruct human 3D pose from depth image using the body part detectors proposed by [9] as regularizing component. Further accuracy improvements were achieved by [1, 20] using regularizing poses from a pre-recorded database as input to the generative tracker. Here, [1] was the first approach running at real-time framerates of more than 50 fps, whereas Ye *et al.*'s method [20] is an offline approach. Other real-time algorithms were proposed by *e.g.* [17] that use a body-part detector similar to [13] to augment a generative tracker. However, none of these hybrid approaches is able to give a meaningful pose hypothesis for non-visible body parts in case of occlusions.

Methods that reconstruct motions based on inertial sensors only have been proposed *e.g.* in [7, 15]. Here, either densely placed sensors or large databases containing motions are used. Also, reconstructing the global position is not possible.

Only a few vision algorithms so far use fusion with complementary sensor systems for full-body tracking. One approach combining 3D inertial information and multi-view markerless motion capture was presented in [10]. Here, the orientation data of five inertial sensors was used as additional energy term to stabilize the local pose optimization. Another example is [23] who fuse information from densely placed inertial sensors is fused with global position estimation using a laser range scanner equipped robot accompany-

[1] http://resources.mpi-inf.mpg.de/InertialDepthTracker

ing the tracked person.

3. Hybrid Inertial Tracker - An Overview

Recent hybrid (generative + discriminative) monocular tracking algorithms *e.g.* [1, 17] can track human skeletons in real-time from a single depth camera, as long as the body is mostly front-facing. However, even in frontal poses, tracking may fail due to complex self-occlusions, limbs close to the body, and other ambiguities. It certainly fails if large sections of the body are completely invisible to the camera, such as in lateral postures, see Fig. 1c. Our new hybrid depth-based tracker succeeds in such cases by incorporating additional inertial sensor data for tracking stabilization. While our concepts are in general applicable to a wide range of generative approaches, discriminative approaches and hybrid approaches, we modify the hybrid depth-based tracker by Baak *et al.* [1] to demonstrate our concepts. This tracker uses discriminative features detected in the depth data, so-called *geodesic extrema* $E_{\mathcal{I}}$, to query a database containing pre-recorded full body poses. These poses are then used to initialize a generative tracker that optimizes skeletal pose parameters $\mathcal{X}$ of a mesh-based human body model $\mathcal{M}_{\mathcal{X}} \subseteq \mathbb{R}^3$ to best explain the 3D point cloud $\mathcal{M}_{\mathcal{I}} \subseteq \mathbb{R}^3$ of the observed depth image $\mathcal{I}$. In a late fusion step, the tracker decides between two pose hypotheses: one obtained using the database pose as initialization or one obtained that used the previously tracked poses as initialization. Baak *et al.*'s approach makes two assumptions: The person to be tracked is facing the depth camera and all body parts are visible to the depth camera, which means it fails in difficult poses mentioned earlier (see Fig. 1 for some examples).

In our new hybrid approach, we overcome these limitations by modifying every step in the original algorithm to benefit from depth and inertial data together. In particular, we introduce a *visibility model* to decide what data modality is best used in each pose estimation step, and develop a discriminative tracker combining both data. We also empower generative tracking to use both data for reliable pose inference, and develop a new late fusion step using both modalities.

Body Model Similar to [1], we use a body model comprising a surface mesh $\mathcal{M}_{\mathcal{X}}$ of 6 449 vertices, whose deformation is controlled by an embedded kinematic skeleton of 62 joints and 42 degrees of freedom via surface skinning. Currently, the model is manually adapted to the actor, but automatic shape adaptation is feasible, see *e.g.* [18]. Furthermore, let $\mathcal{B}_{\text{all}} := \{\text{larm}, \text{rarm}, \text{lleg}, \text{rleg}, \text{body}\}$ be a set of *body parts* representing the left and right arm, left and right leg and the rest of the body. Now, we define five disjoint subsets $\mathcal{M}^b_{\mathcal{X}}, b \in \mathcal{B}_{\text{all}}$ containing all vertices from $\mathcal{M}_{\mathcal{X}}$ belonging to body part b.

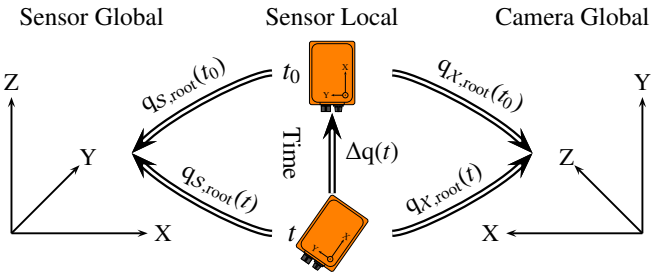

Figure 2. Relationship between the different IMU coordinate systems and orientations.

Sensors As depth camera we use a Microsoft Kinect running at 30 fps, but in Sect. 8 we also show that our approach works on time-of-flight camera data. As additional sensors, we use *inertial measurement units* (IMUs) which are able to determine their relative orientation with respect to a global coordinate system, irrespective of visibility from a camera. IMUs are nowadays manufactured cheaply and compactly, and integrated into many hand-held devices, such as smart phones and game consoles. In this paper, we use six Xsens MTx IMUs, attached to the trunk (s_{root}), the forearms (s_{larm}, s_{rarm}), the lower legs (s_{lleg}, s_{rleg}), and the head (s_{head}), see Fig. 4a. The sensor s_{root} gives us information about the global body orientation, while the sensors on arms and feet give cues about the configuration of the extremities. Finally, the head sensor is important to resolve some of the ambiguities in sparse inertial features. For instance, it helps us to discriminate upright from crouched full body poses. The sensors' orientations are described as the transformations from the sensors' local coordinate systems to a global coordinate system and are denoted by q_{root}, q_{larm}, q_{rarm}, q_{lleg}, q_{rleg}, and q_{head}. In our implementation, we use unit quaternions for representing these transformations, as they best suit our processing steps.

For ease of explanation, we introduce the concept of a *virtual sensor* which provides a simulated orientation reading of an IMU for a given pose $\mathcal{X}$ of our kinematic skeleton. Furthermore, the transformation between the virtual sensor's coordinate system and the depth camera's global coordinate system can be calculated. For clarity, we add $\mathcal{X}$ or $\mathcal{S}$ to the index, *e.g.* $\text{q}_{\mathcal{S},\text{root}}$ denotes the measured orientation of the real sensor attached to the trunk, while $\text{q}_{\mathcal{X},\text{root}}$ represents the readings of the virtual sensor for a given pose $\mathcal{X}$. Note, while the exact placement of the sensors relative to the bones is not so important, it needs to be roughly the same for corresponding real and virtual sensors. Further calibration of the sensors is not required. An orientation of a sensor at time t is denoted as $\text{q}_{\text{root}}(t)$.

4. Visibility Model

Our visibility model enables us to reliably detect global body pose and the visibility of body parts in the depth cam-

era. This information is then used to establish reliable correspondences between the depth image and body model during generative tracking, even under occlusion. Furthermore, it enables us to decide whether inertial or optical data are more reliable for pose retrieval.

Global body position and orientation. In [1], the authors use plane fitting to a heuristically chosen subset of depth data to compute body orientation and translation of the depth centroid. Their approach fails if the person is not roughly facing the camera or body parts are occluding the torso. Inertial sensors are able to measure their orientation in space independent of occlusions and lack of data in the depth channel. We thus use the orientation of the sensor s_{root} to get a good estimate of the body's front direction $\boldsymbol{f}$ within the camera's global coordinate system, even in difficult non-frontal poses, Fig. 3b. However, inertial sensors measure their orientation with respect to some global sensor coordinate system that in general is not identical to the camera's global coordinate system, see also Fig. 2. For that reason, we calculate the transformation $\mathrm{q}_{\mathcal{X},\text{root}}(t)$ in a similar fashion as described in [10] using relative transformations $\Delta\mathrm{q}(t) := \overline{\mathrm{q}_{\mathcal{S},\text{root}}(t_0)} \circ \mathrm{q}_{\mathcal{S},\text{root}}(t)$ with respect to an initial orientation at time t_0. Here, $\overline{\mathrm{q}}$ denotes the inverse transformation of q, while $\mathrm{q}_2 \circ \mathrm{q}_1$ expresses that transformation q_2 is executed after transformation q_1. The transformations $\overline{\mathrm{q}_{\mathcal{S},\text{root}}(t_0)}$ and $\mathrm{q}_{\mathcal{S},\text{root}}(t)$ can be directly obtained from the sensor's measurement. The desired transformation from the sensor's coordinate system to the camera's global coordinate system at time t is now $\mathrm{q}_{\mathcal{X},\text{root}}(t) = \mathrm{q}_{\mathcal{X},\text{root}}(t_0) \circ \Delta\mathrm{q}(t)$. Note that $\mathrm{q}_{\mathcal{X},\text{root}}(t_0)$ can not be measured. Instead, we calculate it using virtual sensors and an initial pose $\mathcal{X}(t_0)$ at time t_0. For this first frame, we determine the front direction $\boldsymbol{f}(t_0)$ as described in [1] and then use our tracker to compute $\mathcal{X}(t_0)$. In all other frames, the front facing direction is defined as

$$\boldsymbol{f}(t) := \mathrm{q}_{\mathcal{X},\text{root}}(t) \circ \overline{\mathrm{q}_{\mathcal{X},\text{root}}(t_0)}[\boldsymbol{f}(t_0)]. \tag{1}$$

Here, $\mathrm{q}[\boldsymbol{v}]$ means that the transformation q is applied to the vector $\boldsymbol{v}$, Fig. 3b.

Body part visibility. The second important information supplied by our visibility model is which parts of the model are visible to the depth camera. To infer body part visibility, we compute all vertices $C_{\mathcal{X}} \subseteq \mathcal{M}_{\mathcal{X}}$ of the body mesh that the depth camera sees in pose $\mathcal{X}$. To this end, we resort to rendering of the model and fast OpenGL visibility testing. Now, the *visibility* of a body part b is defined as

$$V_b := \frac{|\mathcal{M}^b_{\mathcal{X}} \cap C_{\mathcal{X}}|}{|\mathcal{M}^b_{\mathcal{X}}|}. \tag{2}$$

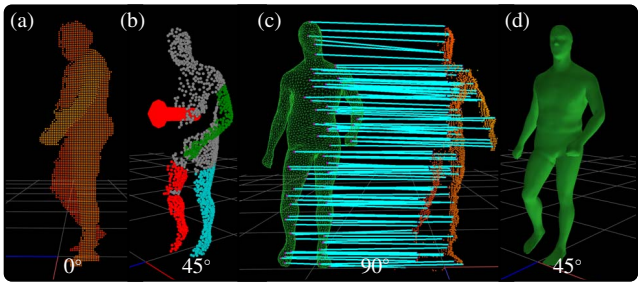

Figure 3. Tracking of frame at 5.0 s of sequence D_6 from our evaluation dataset. The views are rotated around the tracked person, where offset *w. r. t.* the depth camera is depicted at the bottom of each subfigure. **(a)** Input depth data. **(b)** Output of the visibility model. Note: the right arm is not visible. **(c)** Correspondences used by the generative tracker. Note: no correspondences with right arm. The pose parametrized mesh was moved to the left for better visibility. **(d)** Final fused pose.

The set of *visible body parts* is denoted as $\mathcal{B}_{\text{vis}} := \{b \in \mathcal{B}_{\text{all}} : V_b > \tau_3\}$. Note, that the accuracy of $\mathcal{B}_{\text{vis}}$ depends on $\mathcal{M}_{\mathcal{X}}$ resembling the actual pose assumed by the person in the depth image as closely as possible, which is not known before pose estimation. For this reason, we choose the pose $\mathcal{X} = \mathcal{X}^{\text{DB}}$, obtained by the discriminative tracker which yields better results than using the pose $\mathcal{X}(t-1)$ from the previous step, (see Sect. 6). To account for its possible deviation from the "real" pose and to avoid false positives in the set $\mathcal{B}_{\text{vis}}$, we introduce the threshold $\tau_3 > 0$. In the tested scenarios, values of τ_3 up to 10% have shown a good trade-off between rejecting false positives and not rejecting to many body parts, that are actually visible.

In the rendering process also a *virtual* depth image $\mathcal{I}_{\mathcal{X}}$ is created, from which we calculate the first $M = 50$ geodesic extrema in the same way as for the real depth image $\mathcal{I}$, see [1]. Finally, we denote the vertices that generated the extrema's depth points with $C^M_{\mathcal{X}}$.

5. Generative Pose Estimation

Similar to [1], generative tracking optimizes skeletal pose parameters by minimizing the distance between corresponding points on the model and in the depth data. Baak *et al.* fix $C_{\mathcal{X}}$ manually, and never update it during tracking. For every point in $C_{\mathcal{X}}$ they find the closest point in the depth point cloud $\mathcal{M}_{\mathcal{I}}$, and minimize the sum of distances between model and data points by local optimization in the joint angles. Obviously, this leads to wrong correspondences if the person strikes a pose in which large parts of the body are occluded.

In our approach, we also use a local optimization scheme to find a pose $\mathcal{X}$ that best aligns the model $\mathcal{M}_{\mathcal{X}}$ to the point cloud $\mathcal{M}_{\mathcal{I}}$. In contrast to prior work, it also considers which parts of the body are visible and can actually contribute to explaining a good alignment into the depth image. Further-

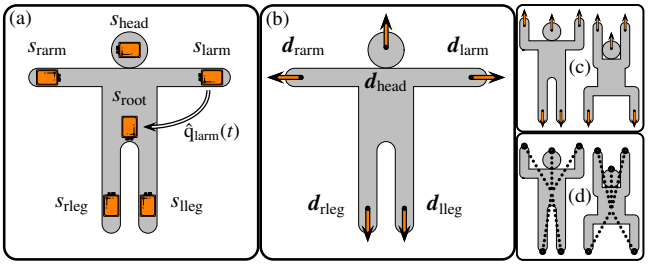

Figure 4. **(a)** Placement of the sensors on the body and normalized orientation *w. r. t.* s_{root}. **(b)** Body part directions used as inertial features for indexing the database. **(c)** Two poses that cannot be distinguished using inertial features. **(d)** The same two poses look different when using optical features.

more, we define subsets $\mathcal{X}_b, b \in \mathcal{B}_{\text{all}}$ of all pose parameters in $\mathcal{X}$ that affect the corresponding point sets $\mathcal{M}_{\mathcal{X}}^b$. We define the set of *active pose parameters* $\mathcal{X}_{\text{act}} := \bigcup_{b \in \mathcal{B}_{\text{vis}}} \mathcal{X}_b$. Finally, the energy function is given as

$$d(\mathcal{M}_{\mathcal{X}}, \mathcal{M}_{\mathcal{I}}) := d_{\mathcal{M}_{\mathcal{X}} \to \mathcal{M}_{\mathcal{I}}} + d_{\mathcal{M}_{\mathcal{I}} \to \mathcal{M}_{\mathcal{X}}} \tag{3}$$

$$d_{\mathcal{M}_{\mathcal{X}} \to \mathcal{M}_{\mathcal{I}}} := \frac{1}{M} \sum_{v \in C_{\mathcal{X}}^M} \min_{p \in \mathcal{M}_{\mathcal{I}}} \|p - v\|_2 \tag{4}$$

$$d_{\mathcal{M}_{\mathcal{I}} \to \mathcal{M}_{\mathcal{X}}} := \frac{1}{N} \sum_{e \in E_{\mathcal{I}}^N} \min_{v \in \mathcal{M}_{\mathcal{X}}} \|e - v\|_2. \tag{5}$$

Here, $E_{\mathcal{I}}^N$ represents the first $N = 50$ geodesic extrema in $\mathcal{I}$, while $C_{\mathcal{X}}^M$ is a subset of $C_{\mathcal{X}}$ containing $M = 50$ visible vertices, see Sect. 4 for details. A visualization for the resulting correspondences can be seen in Fig. 3c. As opposed to Baak *et al.*, we minimize $d(\mathcal{M}_{\mathcal{X}}, \mathcal{M}_{\mathcal{I}})$ using a gradient descent solver similar to the one used in [14] and employ analytic derivatives.

6. Discriminative Pose Estimation

In hybrid tracking, discriminative tracking complements generative tracking by continuous re-initialization of the pose optimization when generative tracking converges to an erroneous pose optimum (see also Sect. 7). We present a new discriminative pose estimation approach that retrieves poses from a database with 50 000 poses obtained from motion sequences recorded using a marker-based mocap system. It adaptively relies on optical features for pose look-up, and new inertial features, depending on visibility and thus reliability of each sensor type. In combination, this enables tracking of poses with strong occlusions, and it stabilizes pose estimation in front-facing poses.

Optical database lookup. In order to retrieve a pose $\mathcal{X}_{\mathcal{I}}^{\text{DB}}$ matching the one in the depth image from the database, Baak *et al.* [1] use geodesic extrema computed on the depth map as index. In their original work, they expect that the first five geodesic extrema $E_{\mathcal{I}}^5$ from the depth image $\mathcal{I}$ are roughly co-located with the positions of the body extrema (head, hands and feet). Geodesic extrema also need to be correctly labeled. Further on, the poses in their database are normalized *w. r. t.* to global body orientation which reduces the database size. As a consequence, also queries into the database need to be pose normalized. We use Baak *et al.*'s geodesic extrema for optical lookup, but use our more robust way for estimating $\boldsymbol{f}(t)$ for normalization, see Sect. 4. Our method thus fares better even in poses where all geodesic extrema are found, but the pose is lateral to the camera.

Inertial database lookup. In poses where not all body extrema are visible, or where they are too close to the torso, the geodesic extrema become unreliable for database lookup. In such cases, we revert to IMU data, in particual their orientations relative to the coordinate system of the sensor s_{root}, see Fig. 4a. Similar to the optical features based on geodesic extrema, these normalized orientations $\hat{\mathrm{q}}_b(t) := \overline{\mathrm{q}_{\text{root}}(t)} \circ \mathrm{q}_b(t)$, $b \in \mathcal{B} = \{\text{larm}, \text{rarm}, \text{lleg}, \text{rleg}, \text{head}\}$ are invariant to the tracked person's global orientation but capture the relative orientation of various parts of the person's body. However, using these normalized orientations directly as index has one disadvantage. This is because many orientation representations need special similarity metrics that are often incompatible to fast indexing structures, such as kd-trees. To this end, we use a vector $\hat{\boldsymbol{d}}_b \in \mathbb{R}^3$ that points in the direction of the bone of a body part, see Fig. 4b. In our setup, these directions are co-aligned with the sensors' local X-axis for all sensors except for the sensor s_{head}, where it is co-aligned with the local Y-axis. The normalized directions $\hat{\boldsymbol{d}}_b(t) := \hat{\mathrm{q}}_b(t)[\boldsymbol{d}_b]$ are then stacked to serve as inertial feature-based query to the database. The retrieved pose is denoted as $\mathcal{X}_{\mathcal{S}}^{\text{DB}}$.

Selecting optical or inertial lookup. At first sight, it may seem that inertial features alone are sufficient to look up poses from the database, because they are independent from visibility issues. However, with our sparse set of six IMUs, the inertial data alone are often not discriminative enough to exactly characterize body poses. Some very different poses may induce the same inertial readings, and are thus ambiguous, see also Fig. 4c. Of course, adding more IMUs to the body would remedy the problem but would starkly impair usablity and is not necessary as we show in the following. Optical geodesic extrema features are very accurate and discriminative of a pose, given that they are reliably found, which is not the case for all extrema in difficult non-frontal starkly occluded poses, see Fig. 4d. Therefore, we introduce two reliability measures to assess the reliability of optical features for retrieval, and use the inertial features only as fall-back modality for retrieval in case optical feautures

cannot be trusted. We use the distances $\epsilon_i(t)$ of the geodesic extrema $i \in \{1, \ldots, 5\}$ at frame t *w. r. t.* the centroid of the point cloud which roughly lies as the center of the torso. For each end effector that distance does not change dramatically across poses in normal motion. When a geodesic extremum is not detected correctly, the computed distance $\epsilon_i(t)$ therefore typically differs significantly from $\bar{\epsilon}_i$. In practice, the distances can be obtained after the first pass of the modified Dijkstra's algorithm, presented in [1]. This yields our first reliability measure

$$\epsilon(t) := \sum_{i=1}^{5} |\epsilon_i(t) - \bar{\epsilon}_i|, \tag{6}$$

The values of $\bar{\epsilon}_i$ for a specific actor are computed once from a short sequence of depth images in which geodesic extrema were detected reliably.

A second reliability measure is the difference between the purely optical computation of the global body pose similar to Baak *et al.* and the inertial sensors measured orientations. More precisely, we use the measure

$$\Delta(t) := \sum_{b \in \mathcal{B}} \delta(\hat{\mathrm{q}}_{\mathcal{X}_I^{\mathrm{DB}},b}(t), \hat{\mathrm{q}}_{\mathcal{S},b}(t)). \tag{7}$$

$\delta = \cos^{-1} |\langle \cdot, \cdot \rangle|$ measures the difference between rotations that we represent as quaternions, where $\langle \cdot, \cdot \rangle$ is the dot product treating quaternions as 4D vectors. The final retrieved pose is computed as

$$\mathcal{X}^{\mathrm{DB}} := \begin{cases} \mathcal{X}_I^{\mathrm{DB}}, & \text{if } \epsilon(t) < \tau_1 \wedge \Delta(t) < \tau_2 \\ \mathcal{X}_S^{\mathrm{DB}}, & \text{otherwise} \end{cases} . \tag{8}$$

We found experimentally that $\tau_1 = 1.15$ and $\tau_2 = 4.0$ are good values for all motion sequences we tested.

7. Final Pose Estimation

The final pose computed by our algorithm is found in a late fusion step. We are running two local pose optimizations (Sect. 5), one using the database pose $\mathcal{X}^{\mathrm{DB}}$ as initialization for the optimizer, and one using the pose from the last frame $\mathcal{X}^{\mathrm{last}}$ as initialization. Here, we are only optimizing for those parameters that are part of $\mathcal{X}_{\mathrm{act}}$. The resulting optimized poses are called $\mathcal{X}_{\mathrm{opt}}^{\mathrm{DB}}$ and $\mathcal{X}_{opt}^{\mathrm{last}}$. From those two, we select the best pose according to Eq. (3). Those parameters that are not part of $\mathcal{X}_{\mathrm{act}}$ are taken over from $\mathcal{X}_S^{\mathrm{DB}}$. This way, even if body parts were occluded or unreliably captured by the camera, we obtain a final result that is based on actual sensor measurements, and not only hypothesized from some form of prior.

8. Results

The C++ implementation of our tracker runs at around 30 fps on a PC with a 2.4 GHz Intel Core i7-2760 QM CPU.

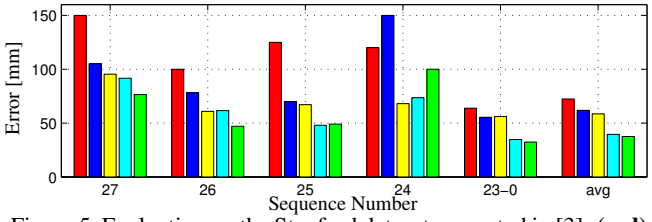

Figure 5. Evaluation on the Stanford dataset presented in [3]. **(red)** Ganapathi *et al.* [3] **(blue)** Baak *et al.* [1] **(yellow)** Our tracker. **(cyan)** Ye *et al.* [20] (not real-time). **(green)** Taylor *et al.* [16].

We qualitatively and quantitatively evaluate it and its components on several data sets and compare to related methods from the literature.

We use a pose database with 50 000 poses. 44 000 were kindly provided by Baak *et al.* [1]. We include 6 000 additional poses that we recorded along with the evaluation data set (Sect. 8.2). These poses show similar types of motion, but are not part of the evaluation set. The pose database is recomputed for each actor once to match his skeleton dimension.

8.1. Evaluation on Stanford Dataset

We evaluate our tracker on the 28 sequences of the Stanford data set from [3]. This data set was recorded with a Swissranger SR 4000 time-of-flight camera and provides ground-truth marker positions from a Vicon motion capture system. However, the data neither contain a pose parameterized model of the recorded person nor inertial sensor data. We therefore estimated the size of the recorded person using a deformable shape model from a set of isulated depth frames obtained from the dataset, see [18] for details. Using the mesh of the fitted model, we designed a suitable skeleton with the same topology as required by our pose parameterized model. We tracked the whole dataset using an IK-tracker and the provided ground-truth marker positions as constraints. The obtained pose parameters were used to compute virtual sensor readings. Note, that there are a lot of manual preprocessing steps involved to make our tracker run on this data set, and each step introduces errors that are not part of the evaluation of the other tested trackers (we copied over their error bars from the respective papers). We now, tracked the dataset using the provided depth frames as well as the virtual sensor readings with our tracker and computed the error metric as described in [3], Fig. 5.

Discussion We used the mean errors according to the error metric described by [3] to compare our tracker to the ones of Ganapathi *et al.* [3], Baak *et al.* [1], Ye *et al.* [20] which is not a real-time tracker, and Taylor *et al.* [16]. Here, we averaged the results of the sequences 0–23 that contain relatively easy to track motions and where our tracker is

performing comparable to previous approaches with little difference across sequences (see additional material for full table). By mean error, our tracker performs better than [3] and [1] on most sequences, and is close to the others on all data (see comments at end). However, our tracker shows its true advantage on sequences with more challenging motion, 24–27, of which only 24 shows notable non-frontal poses, and periods where parts of the body are completely invisible. Here, one can see that other trackers fail, as the errors of most trackers roughly double with respect to the mean error on other sequences. In contrast, our tracker shows an increase of only about 15%, as it continues to follow the motion throughout the sequence. Please note the mean errors are not the best metric to asses our tracker, but are the only values reported in all other papers. The absolute mean errors of our tracker are likely biased by an overhead stemming from the preprocessing mentioned above, and mask its stark improvement on occluded poses.

8.2. Evaluation Dataset

For more reliable testing of our tracker's performance, we recorded a new dataset containing a substantial fraction of challenging non-frontal poses and stark occlusions of body parts. For recording we used one Microsoft Kinect, six Xsens MTx IMUs, as well as a PhaseSpace marker-based optical mocap system with 38 markers. The IMUs were strapped to the head, lower legs, the trunk, and forearms, and are co-align with the assumed virtual sensors, see also Sect. 3. In the following, we assume that all data is temporally aligned and the Kinect data and the marker-based system are spatially aligned. We recorded 6 different sequences ($D_1, \ldots, D_6$) with varying difficulties including punching, kicking, rotating on the spot, sidewards and circular walking performed by one actor (See additional material for details). This totals in about 6 000 frames at 30 Hz. For all sequences we computed ground truth pose parameters and joint positions using the recorded marker positions and the same kinematic skeleton that we use in our tracker. For a qualitative evaluation of our tracker, also in comparison to previous approaches, we refer to Fig. 1 and the accompanying video.

Discussion With this data, we quantitatively compare our tracker (hDB) to the Kinect SDK as well as Baak *et al.*'s work [1]. Note that the tracker of the Kinect SDK is implementing the approach of Shotton *et al.* [13]. We also quantitatively evaluate our tracker with only optical retrieval (oDB), and only inertial retrieval (iDB). To make results of very different trackers comparable, we introduce a new error measure based on joints. Since all trackers use a slightly different set of joints, we select for each tracker a subset of 16 joints that are close to semantic positions in the body such as the lower back, the middle of the back, the upper back, the head, the shoulders, the elbows, the wrists, the hips, the knees, and the ankles. Furthermore, as the corresponding joints from the different trackers do not lie at the exact same positions we need to normalize for this offset. We do this by calculating the average local displacement (*i. e.* local within the frame of the ground truth joint) of the joint relative to the corresponding ground-truth joint, and subtracting this offset from the position of the tracked joint.

Fig. 6a shows the average joint error for all tested trackers and algorithm variants on all 6 sequences. On the first four sequences which are easier and show no non-frontal poses, our final tracker (hDB) is among the best ones and, as expected, mostly comparable to Ganapathi's and Baak's methods. Importantly, it is always better than iDB and oDB. However, hDB outperforms all other approaches on the last two sequences, *e.g.* producing less than half the error (about 75 mm) of Baak *et al.* [1] with about 180 mm. The temporal error evolution of some representative joints in D_5 and D_6 are depicted in Fig. 6b for Kinect SDK, Baak *et al.*, and our algorithm. This clearly shows that our algorithm produces significantly lower errors than the other trackers on certain spans of poses, which is masked in average error values. Finally, Fig. 6c shows the superiority of our tracker on selected time steps from that sequence, by visually comparing each result to ground truth joint locations (see video for more results). Error plots for the other joints and sequences can be found in the supplemental material. Here, we also included errors of our tracker, where one of the database-lookup strategies—either the optical or the inertial—was deactivated to show its impact on the overall performance. Our final tracker also performs consistently better than iDB and oDB illustrating the benefit of our fusion strategy. This is particularly evident in D_3 and D_4. Sequence D_3 contains squats, on which inertial feature lookup is ambiguous. D_4 contains motions where the arms touch the body at different locations. Here, the database lookup based on optical features fails.

9. Conclusions

We presented a hybrid method to track human full body poses from a single depth camera and additional inertial sensors. Our algorithm runs in real-time and, in contrast to previous methods, captures the true body configuration even in difficult non-frontal poses and poses with partial and substantial visual occlusions. The core of the algorithm are new solutions for depth and inertial data fusion in a combined generative and discriminative tracker. We have demonstrated our tracker's performance qualitatively and quantitatively on a large corpus of data that we provide to the community, and showed its clear advantages over other state-of-the-art methods.

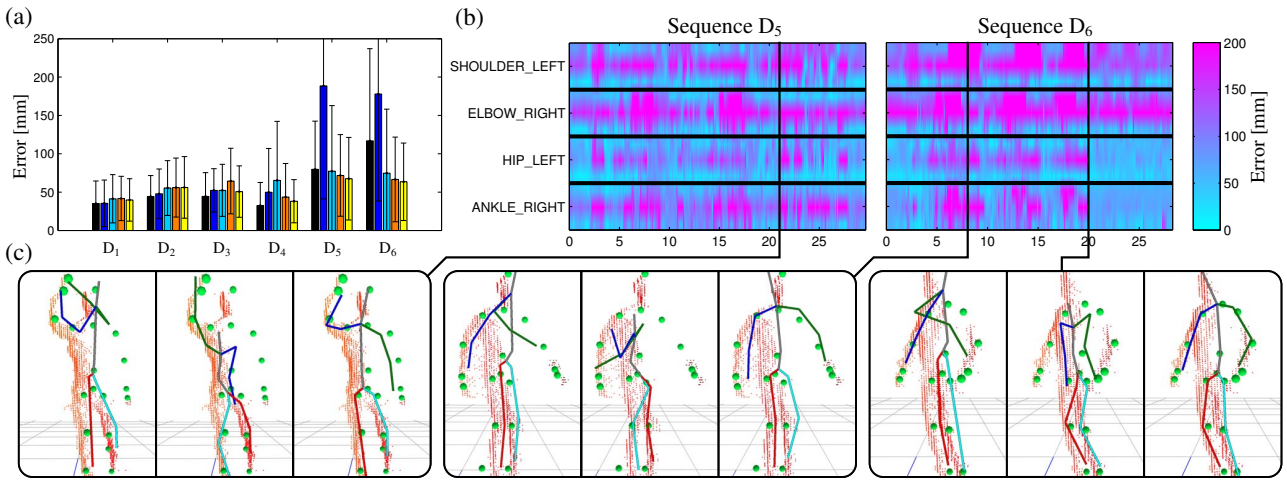

Figure 6. **(a)** Average joint tracking error in millimeters for sequences $D_1, \ldots, D_6$ from our evaluation dataset, tracked with Kinect SDK's joint tracker **(black)**, Baak *et al.* **(blue)**, and our tracker with only optical DB lookup (oDB) **(light blue)**, only inertial DB lookup (iDB) **(orange)**, and the proposed combined DB lookup (hDB) **(yellow)**. **(b)** Joint errors for selected joints over sequences D_5 and D_6 (time in seconds). Per joint there are three error rows: **(top)** Kinect SDK's tracker, **(middle)** Baak *et al.*, and **(bottom)** our approach. **(c)** Three challenging example poses from sequences D_5 and D_6. Input depth data, ground-truth joint positions **(green dots)** and tracked **(skeleton)** shown from the side. Our approach **(right)** clearly outperforms the Kinect SDK's tracker **(left)**, and Baak *et al.*'s method **(middle).**

References

[1] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *ICCV*, 2011.

[2] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real-time human pose tracking from range data. In *ECCV*, 2012.

[3] V. Ganapathi, C. Plagemann, S. Thrun, and D. Koller. Real time motion capture using a single time-of-flight camera. In *CVPR*, 2010.

[4] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *ICCV*, pages 415–422, 2011.

[5] S. Knoop, S. Vacek, and R. Dillmann. Fusion of 2D and 3D sensor data for articulated body tracking. *Robotics and Autonomous Systems*, 57(3):321–329, 2009.

[6] A. Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight sensors in computer graphics. *CGF*, 29(1):141–159, 2010.

[7] H. Liu, X. Wei, J. Chai, I. Ha, and T. Rhee. Realtime human motion control with a small number of inertial sensors. In *I3D*, pages 133–140, 2011.

[8] Y. Pekelny and C. Gotsman. Articulated object reconstruction and markerless motion capture from depth video. *CGF*, 27(2):399–408, 2008.

[9] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. Real-time identification and localization of body parts from depth images. In *ICRA*, Anchorage, Alaska, USA, 2010.

[10] G. Pons-Moll, A. Baak, T. Helten, M. Müller, H.-P. Seidel, and B. Rosenhahn. Multisensor-fusion for 3d full-body human motion capture. In *CVPR*, pages 663–670, 2010.

[11] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[12] M. Salzmann and R. Urtasun. Combining discriminative and generative methods for 3D deformable surface and articulated pose reconstruction. In *CVPR*, 2010.

[13] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *CVPR*, 2011.

[14] C. Stoll, N. Hasler, J. Gall, H.-P. Seidel, and C. Theobalt. Fast articulated motion tracking using a sums of gaussians body model. In *ICCV*, pages 951–958, 2011.

[15] J. Tautges, A. Zinke, B. Krüger, J. Baumann, A. Weber, T. Helten, M. Müller, H.-P. Seidel, and B. Eberhardt. Motion reconstruction using sparse accelerometer data. *TOG*, 30(3):18:1–18:12, 2011.

[16] J. Taylor, J. Shotton, T. Sharp, and A. W. Fitzgibbon. The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR*, 2012.

[17] X. Wei, P. Zhang, and J. Chai. Accurate realtime full-body motion capture using a single depth camera. *TOG*, 31(6):188:1–188:12, 2012.

[18] A. Weiss, D. Hirshberg, and M. Black. Home 3D body scans from noisy image and range data. In *ICCV*, 2011.

[19] G. Ye, Y. Liu, N. Hasler, X. Ji, Q. Dai, and C. Theobalt. Performance capture of interacting characters with handheld kinects. In *Proc. ECCV*, pages 828–841, 2012.

[20] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *ICCV*, pages 731–738, 2011.

[21] L. Zhang, B. Curless, and S. M. Seitz. Spacetime stereo: Shape recovery for dynamic scenes. In *CVPR*, 2003.

[22] Y. Zhu, B. Dariush, and K. Fujimura. Kinematic self retargeting: A framework for human pose estimation. *CVIU*, 114(12):1362–1375, 2010.

[23] J. Ziegler, H. Kretzschmar, C. Stachniss, G. Grisetti, and W. Burgard. Accurate human motion capture in large areas by combining IMU- and laser-based people tracking. In *IROS*, pages 86–91, 2011.