Capturing Hand Motion with an RGB-D Sensor, Fusing a Generative Model with Salient Points

Dimitrios Tzionas[1,2], Abhilash Srikantha[1,2], Pablo Aponte[2], and Juergen Gall[2]

[1] Perceiving Systems Department, MPI for Intelligent Systems, Germany
dimitris.tzionas@tue.mpg.de, abhilash.srikantha@tue.mpg.de
[2] Computer Vision Group, University of Bonn, Germany
aponte@iai.uni-bonn.de, gall@iai.uni-bonn.de

Abstract. Hand motion capture has been an active research topic, following the success of full-body pose tracking. Despite similarities, hand tracking proves to be more challenging, characterized by a higher dimensionality, severe occlusions and self-similarity between fingers. For this reason, most approaches rely on strong assumptions, like hands in isolation or expensive multi-camera systems, that limit practical use. In this work, we propose a framework for hand tracking that can capture the motion of two interacting hands using only a single, inexpensive RGB-D camera. Our approach combines a generative model with collision detection and discriminatively learned salient points. We quantitatively evaluate our approach on 14 new sequences with challenging interactions.

1 Introduction

Human body tracking has been a popular field of research during the past decades [25], recently gaining more popularity due to the ubiquity of RGB-D sensors. Hand motion capture, a special instance of it, has enjoyed much research interest [11] due to its numerous applications including, but not limited to, computer graphics, human-computer-interaction and robotics.

Despite similarities, robust techniques [40] for full-body tracking are insufficient for hand motion capture, as the latter is more complicated on numerous fronts. Hands are characterized by more degrees of freedom, formulating a higher dimensional optimization problem. Severe occlusions are a usual phenomenon, being either self-occlusions or occlusions from another hand or object. Similarity in shape and appearance causes ambiguities for the differentiation between fingers and hands. Fast motion and lower resolution of hands in images constitute further complicating factors.

Despite theses challenges, there has been substantial progress in hand motion capture in recent years. Ballan et al. [3] have presented a system that successfully captures the motion of two hands strongly interacting with each other and an additional object. Although the approach achieves remarkable accuracy, it is based on an expensive and elaborate multi-camera system. On the other hand, Oikonomidis et al. [27–29] have presented a real time hand tracker using just a single off-the-shelf RGB-D camera. Despite their success under challenging scenarios, the exhibited accuracy is not as precise as [3].

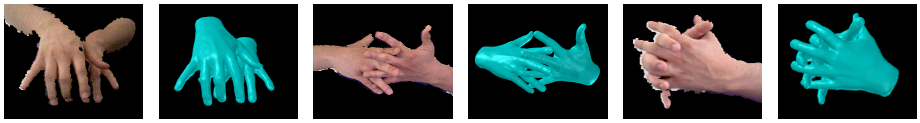

Fig. 1. Qualitative results of our pipeline. Each pair shows the aligned RGB and depth input maps after depth thresholding, along with the pose estimate output

Our approach for tracking the pose of two strongly interacting hands is inspired by Ballan et al. [3] and combines a generative model with an occlusion handling method and a discriminatively trained detector for salient points. While [3] relies on an expensive capture setup with 8 synchronized and calibrated RGB cameras recording FullHD footage at 50 fps, we propose an approach that captures hand motion of two interacting hands using a cheap RGB-D camera recording VGA resolution at 30 fps. We evaluate our approach on 14 annotated sequences[1], which include interactions between hands. We further compare our method to the single hand tracker [27] on sequences with one hand, showing that our approach estimates the hand pose with higher accuracy than [27].

2 Related Work

The study of hand motion tracking has its roots in the 90s [32,33]. Although the problem can be simplified by means of data-gloves [10], color-gloves [48], markers [47] or wearable sensors [21], the ideal solution pursued is the unintrusive, marker-less capture of hand motion. Even until recently the study was mainly confined to the case of a single isolated hand [2,17,24,27,41,42,45]. However, in pursuit of more realistic scenarios, research effort was directed towards the case of a hand interacting with an object [15,16,28], two hands interacting with each other [3,29] and with an additional object [3]. Multiple objects can be tracked by means of hand tracking and physical forces modeling [22].

An analytical review of the field can be found in the work of Erol et al. [11]. In this work a taxonomy is presented, separating the methods met in the literature in two main categories, namely *model-based* and *appearance-based*.

Generative-model approaches [17,31,42] are based on an explicit model used to generate pose hypotheses, which are evaluated against the observed data. The evaluation is based on an objective function which implicitly measures the likelihood by computing the discrepancy between the pose estimate (hypothesis) and the observed data in terms of an error metric. To keep the problem tractable, each iteration is initialized by the pose estimate of the previous step, relying thus heavily on temporal continuity and being prone to accumulative error. The objective function is evaluated in the high-dimensional, continuous parameter space.

[1]The annotated dataset sequences and the supplementary material are available at `http://files.is.tue.mpg.de/dtzionas/GCPR_2014.html`.

Discriminative methods learn a direct mapping from the observed image features to the discrete [2, 34, 35] or continuous [7, 20, 36, 40] target parameter space. Most methods operate on a single frame [2, 7, 36, 40], being thus immune to pose-drifting due to error accumulation. Generalization in terms of capturing illumination, articulation and view-point variation, can be realized only through adequate representative training data. Acquisition and annotation of realistic training data is though a cumbersome and costly procedure. For this reason most approaches rely on synthetic rendered data [20, 35, 40] that has inherent ground-truth. However, the discrepancy between realistic and synthetic data is an important limiting factor, while special care is needed to avoid over-fitting to the training set. The accuracy of discriminative methods heavily depends on the invariance, repeatability and discriminative properties of the features employed and is lower in comparison to generative methods.

A discriminative method can effectively complement a generative method, either in terms of initialization or recovery, driving the optimization framework away from local minima in the search space and aiding convergence to the global minimum. Sridhar et al. [41] combine in a real time system a Sums-of-Gaussians generative model with a discriminatively trained fingertip detector in depth images using a linear SVM classifier. Ballan et al. [3] present an accurate offline tracker that combines in a single framework a generative model with a salient-point (finger-nail) Hough-forest [14] detector in color images. Both approaches [3, 41], however, require an expensive multi-camera hardware setup.

3 Tracking Method

3.1 Hand Model

We resort to the *Linear Blend Skinning* (LBS) model [23], consisting of a triangular mesh, an underlying kinematic skeleton and a set of skinning weights. In our experiments, a triangular mesh of a pair of hands was obtained by a commercial 3D scanning solution and the skeleton structure was manually defined. Additional details are provided in the supplementary material[1]. The skinning weight $\alpha_{\mathbf{v},j}$ defines the influence of bone j on 3D vertex $\mathbf{v}$, where $\sum_j \alpha_{\mathbf{v},j} = 1$. The deformation of the mesh is driven by the underlying skeleton with pose parameter vector θ through the skinning weights and is expressed by the LBS operator:

$$\mathbf{v}(\theta) = \sum_j \alpha_{\mathbf{v},j} T_j(\theta) T_j(0)^{-1} \mathbf{v}(0) \tag{1}$$

where $T_j(0)$ and $\mathbf{v}(0)$ are the bone transformations and vertex positions at the known rigging pose. The skinning weights are computed using [4].

The global rigid motion is represented by a twist [6, 26, 31] in the special Euclidean group $SE(3)$. The articulation of the skeleton is expressed by a kinematic chain of rigid components. For the sake of simplicity, joints with more than 1 degree of freedom (DoF) are modeled by a combination of revolute joints. Using the exponential map operator, the transformation of a bone $T_j(\theta)$ with k DoF

is therefore given by $T_j(\theta) = \prod_{i<j} T_i(\theta) \prod_{k_j} \exp(\theta_{k_j} \hat{\xi}_{k_j})$, where θ is the parameterization of the full pose, $\theta_{k_j} \hat{\xi}_{k_j}$ is the twist representation of a single revolute joint, and $T_{i<j}$ denotes all previous bones in the kinematic chain.

In our experiments, a single hand consists of 31 revolute joints, i.e. 37 DoF. Thus, for sequences with two interacting hands we have to estimate all 74 DoF.

Anatomically inspired joint-angle limits [1] constrain the solution space to the subspace of physically plausible poses as in [3].

3.2 Optimization

Our objective function for pose estimation consists of four terms:

$$\begin{aligned} E(\theta, D) = \quad & E_{model \to data}(\theta, D) + E_{data \to model}(\theta, D) + \\ & E_{salient}(\theta, D) + \gamma_c E_{collision}(\theta) \end{aligned} \tag{2}$$

where θ are the pose parameters of the two hands and D is the current preprocessed depth image. The first two terms minimize the alignment error of the transformed mesh and the depth data. The alignment error is measured by $E_{model \to data}$, which measures how well the model fits the observed depth data, and $E_{data \to model}$, which measures how well the depth data is explained by the model. The last two terms are inspired by [3]. $E_{salient}$ measures the consistency of the generative model with detected salient points in the image. The main purpose of the term in our framework is to recover from tracking errors of the generative model. In our scenario with a single camera of low resolution, the 3D positions of the detected points are less accurate and additional care is needed. $E_{collision}$ penalizes intersections of fingers, ensuring physically plausible poses.

The objective Equation (2) is minimized by local optimization as described in [31]. In the following, we give details for the terms of the objective function.

3.2.1 Preprocessing: For pose estimation, we first remove irrelevant parts of the RGB-D image by thresholding the depth values and applying skin color segmentation [19] on the RGB image. As a result, we get a masked RGB-D image, which is denoted as D in Equation (2). The thresholding of the depth image avoids unnecessary processing like normal computation for points far away and the skin color segmentation removes occluding objects from the data.

3.2.2 Fitting the *model to the data*: The first term in Equation (2) aims at fitting the mesh parameterized by pose parameters θ to the preprocessed data D. To this end, the depth values are converted into a 3D point cloud based on the calibration data of the sensor. The point cloud is then smoothed by a bilateral filter [30] and normals are computed [18]. For each vertex of the model $\mathbf{v}_i(\theta)$, with normal $\mathbf{n}_i(\theta)$, we search for the closest point X_i in the point cloud. This gives a 3D-3D correspondence for each vertex. We discard the correspondence if the angle between the normals of the vertex and the closest point is larger than 45° or the distance between the points is larger than 10 mm. We can then write the term $E_{model \to data}$ as a least squared error of *point-to-point* distances:

$$E_{model \to data}(\theta, D) = \sum_i \| \mathbf{v}_i(\theta) - X_i \|^2 \tag{3}$$

An alternative to the *point-to-point* distance is the *point-to-plane* distance, which is commonly used for 3D reconstruction [9, 38, 39]:

$$E_{model \to data}(\theta, D) = \|\mathbf{n}_i(\theta)^T(\mathbf{v}_i(\theta) - X_i)\|^2 \tag{4}$$

The two distance metrics are evaluated in our experiments (Section 4.1.3). In general, the *point-to-plane* distance converges faster and is therefore preferred.

3.2.3 Fitting the *data to the model*: Only fitting the model to the data is not sufficient as we will show in our experiments. In particular, poses with self-occlusions can have a very low error since the measure only evaluates how well the visible part of the model fits the point cloud. The second term $E_{data \to model}(\theta, D)$ matches the data to the model to make sure that the solution is not degenerate and explains the data as well as possible. Since matching the data to the model is expensive, we reduce the matching to depth discontinuities [13]. To this end, we extract depth discontinuities from the depth map and the projected depth profile of the model using an edge detector [8]. Correspondences are again established by searching for the closest points, but now in the depth image using a 2D distance transform [12]. Similar to $E_{model \to data}(\theta, D)$, we discard correspondences with a large distance. The depth values at the depth discontinuities in D, however, are less reliable not only due to the depth ambiguities between foreground and background, but also due to the noise of cheap sensors. The depth of the point in D is therefore computed as average in a local $3x3$ pixels neighborhood and the outlier distance threshold is increased to 30 mm. The approximation is sufficient for discarding outliers, but insufficient for minimization. For each matched point in D we therefore compute the projection ray uniquely expressed as a Plücker line [31, 37, 43] with direction $\mathbf{d}_i$ and moment $\mathbf{m}_i$ and minimize the least square error between the projection ray and the vertex $\mathbf{v}_i(\theta)$ for each correspondence:

$$E_{data \to model}(\theta, D) = \sum_i \|\mathbf{v}_i(\theta) \times \mathbf{d}_i - \mathbf{m}_i\|^2 \tag{5}$$

3.2.4 Collision detection Collision detection is based on the observation that two objects cannot share the same space and is of high importance in case of self-penetration, inter-finger penetration or general intensive interaction.

For detecting collisions, we use *bounding volume hierarchies* (BVH) to efficiently determine collisions between meshes [44]. Having found a collision between two triangles f_s and f_t, the amount of penetration can be computed as in [3] using a 3D distance field in the form of a cone. Considering the case where the vertices of f_s are the *intruders* and the triangle f_t is the *receiver* of the penetration (similarly for the opposite case), the cone for computing the 3D distance field Ψ_{f_t} is defined by the circumcenter of the triangle f_t. Letting $\mathbf{n}_{f_t}$ denote the normal of the triangle, $\mathbf{o}_{f_t}$ the circumcenter, and r_{f_t} the radius of the circumcircle, we have

$$\Psi_{f_t}(\mathbf{v}_s) = \begin{cases} |(1 - \Phi(\mathbf{v}_s))\Upsilon(\mathbf{n}_{f_t} \cdot (\mathbf{v}_s - \mathbf{o}_{f_t}))|^2 & \text{when} \quad \Phi(\mathbf{v}_s) < 1 \\ 0 & \text{when} \quad \Phi(\mathbf{v}_s) \geq 1 \end{cases} \tag{6}$$

$$\Phi(\mathbf{v}_s) = \frac{\left\|(\mathbf{v}_s - \mathbf{o}_{f_t}) - (\mathbf{n}_{f_t} \cdot (\mathbf{v}_s - \mathbf{o}_{f_t}))\mathbf{n}_{f_t}\right\|}{-\frac{r_{f_t}}{\sigma}(\mathbf{n}_{f_t} \cdot (\mathbf{v}_s - \mathbf{o}_{f_t})) + r} \tag{7}$$

$$\Upsilon(x) = \begin{cases} -x + 1 - \sigma & \text{when} \quad x \le -\sigma \\ -\frac{1-2\sigma}{4\sigma^2}x^2 - \frac{1}{2\sigma}x + \frac{1}{4}(3 - 2\sigma) & \text{when} \quad -\sigma \le x \le +\sigma \\ 0 & \text{when} \quad x \ge +\sigma \end{cases} \tag{8}$$

The parameter σ defines the field of view of the cone and is fixed to 0.5 as in [3].

For each vertex penetrating a triangle, a force can be computed that pushes the vertex back, where the direction is given by the inverse normal direction of the vertex and the strength of the force by Ψ. Using *point-to-point* distances (3), the forces are computed for the set of colliding triangles $\mathcal{C}$:

$$E_{collision}(\theta) = \sum_{(f_s(\theta), f_t(\theta)) \in \mathcal{C}} \left\{ \sum_{\mathbf{v}_s \in f_s} \| - \Psi_{f_t}(\mathbf{v}_s)\mathbf{n}_s \|^2 + \sum_{\mathbf{v}_t \in f_t} \| - \Psi_{f_s}(\mathbf{v}_t)\mathbf{n}_t \|^2 \right\} \tag{9}$$

Though not explicitly denoted, f_s and f_t depend on θ and therefore also Ψ, $\mathbf{v}$ and $\mathbf{n}$. For *point-to-plane* distances (4), the equation gets simplified since $\mathbf{n}^T\mathbf{n} = 1$:

$$E_{collision}(\theta) = \sum_{(f_s(\theta), f_t(\theta)) \in \mathcal{C}} \left\{ \sum_{\mathbf{v}_s \in f_s} \| - \Psi_{f_t}(\mathbf{v}_s) \|^2 + \sum_{\mathbf{v}_t \in f_t} \| - \Psi_{f_s}(\mathbf{v}_t) \|^2 \right\} \tag{10}$$

This term takes part in the objective function (2) regulated by weight γ_c. An evaluation of different γ_c values is presented in our experiments (Section 4.1.1).

3.2.5 Salient point detection: Our approach is so far based on a generative model, which generally provides accurate solutions, but recovers only slowly from ambiguities and tracking errors. It has been shown in [3] that this can be compensated by integrating a discriminatively trained salient point detector into a generative model. In [3], a finger nail detector was applied to the high resolution images and due to the multi-camera setup it could be assumed that the nails become visible in some of the cameras. For low-resolution video of a single camera, finger nails cannot be reliably detected. Instead we train a fingertip detector [14] on raw depth data where the training data is not part of the test sequences. More details are given in the supplementary material.

Since we resort to salient points only for additional robustness, it is usually sufficient to have only sparse fingertip detections. We therefore collect detections with a high confidence, choosing a threshold of $c_{thr} = 3.0$ for our experiments. The association between detections and fingertips of the model, as shown in Table 1, is solved by integer programming [3, 5]:

$$\begin{aligned} \text{argmin} \quad & \sum_{s,t} e_{st} w_{st} + \lambda \sum_s \alpha_s w_s + \lambda \sum_t \beta_t \\ \text{subject to} \quad & \sum_s e_{st} + \beta_t = 1 \qquad \forall t, \\ & \sum_t e_{st} + \alpha_s = 1 \qquad \forall s \\ & e_{st}, \alpha_t, \beta_s \in \{0, 1\} \end{aligned} \tag{11}$$

Table 1. The graph contains T mesh fingertips ξ_t and S fingertip detections δ_s. The cost of assigning a detection δ_s to a finger ξ_t is given by w_{st} as shown in table (a). The cost of declaring a detection as false positive is λw_s where w_s is the detection confidence. The cost of not assigning any detection to finger ξ_t is given by λ. The binary solution of table (b) is constrained to sum up to 1 for each row and column

(a)

		Fingertips ξ_t				V
		ξ_1	ξ_2	$\dots$	ξ_T	α
Detections δ_s	δ_1	w_{11}	w_{12}	$\dots$	w_{1T}	λw_1
	δ_2	w_{21}	w_{22}	$\dots$	w_{2T}	λw_2
	δ_3	w_{31}	w_{32}	$\dots$	w_{3T}	λw_3
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	δ_S	w_{S1}	w_{S2}	$\dots$	w_{ST}	λw_S
V	β	λ	λ	$\dots$	λ	∞

(b)

		Fingertips ξ_t				V
		ξ_1	ξ_2	$\dots$	ξ_T	α
Detections δ_s	δ_1	e_{11}	e_{12}	$\dots$	e_{1T}	α_1
	δ_2	e_{21}	e_{22}	$\dots$	e_{2T}	α_2
	δ_3	e_{31}	e_{32}	$\dots$	e_{3T}	α_3
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	δ_S	e_{S1}	e_{S2}	$\dots$	e_{ST}	α_S
V	β	β_1	β_2	$\dots$	β_T	0

The weights w_{st} are given by the 3D distance between the detection δ_s and the finger of the model ξ_t. For each finger ξ_t, a set of vertices are marked in the model. The distance is then computed as the centroid of the visible vertices of ξ_t and the centroid of the detected region δ_s. For the weights w_s, we investigate two approaches. The first approach uses $w_s = 1$ as in [3]. The second approach takes the confidences c_s of the detections into account by setting $w_s = \frac{c_s}{c_{thr}}$. The weighting parameter λ is evaluated in the experimental section (Section 4.1.2).

If a detection δ_s has been associated to a finger ξ_t, we have to define correspondences between the set of visible vertices of ξ_t and the point cloud of the detection δ_s. If the finger is already very close and the distance is below 10 mm, we do not compute any correspondences since the localization accuracy of the detector is not higher. In this case the finger is anyway close enough to the data to achieve a good alignment. Otherwise, we compute the closest points between the vertices $\mathbf{v}_i$ and the points X_i of the detection:

$$E_{salient}(\theta, D) = \sum_{s,t} e_{st} \left\{ \sum_{(X_i, \mathbf{v}_i) \epsilon \delta_s \times \xi_t} \|\mathbf{v}_i(\theta) - X_i\|^2 \right\} \tag{12}$$

As in (4), a *point-to-plane* distance metric can replace the *point-to-point* metric. When the overlap of the fingertip and the detection is less than 50%, replacing the closest points X_i by the centroid of the detection leads to a speed up.

4 Experimental Evaluation

Benchmarking in the context of 3D hand tracking remains an open problem [11] despite recent contributions [41, 46]. Related RGB-D methods [27] usually report quantitative results only on synthetic sequences, which inherently include ground-truth, while for realistic conditions they resort to qualitative results.

Although qualitative results are informative, quantitative evaluation based on ground-truth is of high importance. We therefore manually annotate 14 new sequences, 11 of which are used to evaluate the components of our pipeline and 3 for comparison with the state-of-the-art method [27] (details in the supplementary material). The standard deviation for 4 annotators is 1.46 pixels.

Table 2. Evaluation of collision weights γ_c, using a 2D distance error metric (px). Weight 0 corresponds to deactivated collision detection, noted as "$LO + S$" in Table 5. Sequences are grouped (see supplementary material) in 3 categories: "*Severe*" for intense, "*some*" for light and "*no apparent*" for imperceptible collision. Our highlighted decision is based on the union of the "*severe*" and "*some*" sets, noted as "*at least some*"

γ_c	0	1	2	3	5	7.5	10	12.5
All	5.40	5.50	5.63	5.23	5.18	5.19	5.18	5.21
Only Severe	6.00	6.18	6.37	5.73	5.66	5.67	5.65	5.71
Only Some	3.99	3.98	3.98	3.98	3.98	3.99	3.99	3.98
At least some	5.52	5.65	5.80	5.31	5.26	5.27	5.25	5.29

Table 3. Evaluation of the parameter λ of the assignment graph of Equation (11), using a 2D distance error metric (px). Value $\lambda = 0$ corresponds to deactivation of the detector, noted as "$LO + C$" in Table 5. Both versions of w_s described in Section 3.2.5 are evaluated

λ	*0*	0.3	0.6	0.9	1.2	1.5	1.8
$w_s = 1$	5.24	5.23	5.21	5.21	5.18	5.19	5.30
$w_s = \frac{c_s}{c_{thr}}$		5.21	5.19	5.18	5.18	5.28	5.68

The error metric for our experiments is the 2D distance (pixels) between the projection of the 3D joints and the corresponding 2D annotations. Details regarding the joints taken into account in the metric are included in the supplementary material. We report the average error over all frames of all sequences.

4.1 Pipeline Components

Our system is based on an objective function consisting of four terms, described in Section 3.2. Two of them minimize the error between the posed mesh and the depth data by fitting the *model to the data* and the *data to the model*. A *salient point* detector further constraints the pose using fingertip detections in the depth image, while a *collision detection* method contributes to realistic pose estimates that are physically plausible. The above terms participate in the objective function in a weighted scheme, which is minimized as in [31].

In the following, we evaluate the parameters used in our components and assess both each component's individual contribution to the overall system performance, as well as of the combination thereof in the objective function (2).

4.1.1 Collision Detection The collision detection component is regulated in the objective function (2) by the weight γ_c, so that collision and penetration get efficiently penalized. Table 2 summarizes our evaluation experiments for the values of γ_c. Although we choose $\gamma_c = 10$ for the present dataset, a generally proposed range of values for new sequences would be between 5 and 10.

4.1.2 Salient Point Detection The salient point detection component depends on the parameters w_s and λ, as described in Section 3.2.5. Table 3 summarizes our evaluation of the parameter λ spanning a range of possible values for both cases $w_s = 1$ and $w_s = \frac{c_s}{c_{thr}}$. Although the difference between the two

Table 4. Evaluation of *point-to-point* (*p2p*) and *point-to-plane* (*p2plane*) distance metrics, along with iterations number of the optimization framework, using a 2D distance error metric (px)

Iterations	5	10	15	20	30
p2p	7.39	5.31	5.11	5.04	4.97
p2plane	5.39	5.18	5.14	5.13	5.11

Table 5. Evaluation of the components of our pipeline. "*LO*" stands for local optimization and includes fitting both *data-to-model* (*d2m*) and *model-to-data* (*m2d*), unless otherwise specified. Collision detection is noted as "*C*", while salient point detector is noted as "*S*". The number of sequences where the optimization framework collapses is noted in the last row, while the mean error is reported only for the rest

Components	LO_{m2d}	LO_{d2m}	*LO*	$LO + C$	$LO + S$	$LO + CS$
Mean Error (px)	15.19	–	5.59	5.24	5.40	5.18
Improvement (%)			–	6.39	3.40	7.36
Failed Sequences	1/11	11/11	0/11	0/11	0/11	0/11

versions of w_s is not very large, the latter performs better for a wide range (0.6 to 1.2) of parameter λ. We therefore choose $w_s = \frac{c_s}{c_{thr}}$ and $\lambda = 1.2$.

4.1.3 Distance Metrics Table 4 presents an evaluation of the two distance metrics presented in Section 3.2.2, namely *point-to-point* (Equation (3)) and *point-to-plane* (Equation (4)), along with the number of iterations of the minimization framework. The *point-to-plane* metric leads to adequate minimization with only 10 iterations, providing a significant speed gain, being thus our choice. However, we perform 50 iterations for the first frame in order to ensure an accurate refinement of the manually initialized pose. The runtime for the chosen setup (see supplementary material for benchmark details) is 2.74 and 4.35 seconds per frame for scenes containing one and two hands respectively.

4.1.4 Component Evaluation Table 5 presents the evaluation of each component and the combination thereof. Simplified versions of the pipeline, fitting either just the *model to the data* or the *data to the model* can lead to a collapse of the pose estimation, due to unconstrained optimization. Our experiments quantitatively show the notable contribution of both the collision detection and the salient point detector components. The best overall system performance is achieved with the combinatorial setup described by the objective function (2). Qualitative results are depicted in Figure 1.

4.2 Comparison to State-of-the-Art

Recently, Oikonomidis et al. used particle swarm optimization (PSO) for a real-time hand tracker [27–29]. These works constitute the state-of-the-art for single-view RGB-D hand tracking. For comparison we use the software released for tracking one hand [27], with the parameter setups of all the above works. Each setup is evaluated 3 times in order to compensate for the manual initialization and the inherent randomness of PSO. Qualitative results depict the best version,

Table 6. Comparison of our method against the FORTH tracker. We evaluate the FORTH tracker with 4 parameter-setups met in the referenced literature of the last column

		Mean (px)	*St.Dev (px)*	*Max (px)*	*Generations*	*Particles*	*Reference*
FORTH	*set 1*	8.58	5.74	61.81	25	64	[27]
	set 2	8.32	5.42	57.97	40	64	[28]
	set 3	8.09	5.00	38.90	40	128	[3]
	set 4	8.16	5.18	39.85	45	64	[29]
Proposed		3.76	2.22	19.92			

while quantitative results report the average. Figure 2 qualitatively showcases the increased accuracy of our method, along with the decreased accuracy of the FORTH tracker due to the sampling nature of PSO. Quantitative results of Table 6 show that our system outperforms [27] in terms of tracking accuracy. However, it should be noted that the GPU implementation of [27] is real time, in contrast to our CPU implementation. Detailed results for each evaluation of the parameter setups are included in the supplementary material.

5 Conclusion

In this work we have presented a system capturing the motion of two highly interacting hands. Inspired by the recent method [3], we propose a combination of a generative model with a discriminatively trained salient point detector and collision modeling to obtain accurate, realistic pose estimates with increased immunity to ambiguities and tracking errors. While [3] depends on expensive, specialized equipment, we achieve accurate tracking results using only a single cheap, off-the-shelf RGB-D camera. We have evaluated our approach on 14 new challenging sequences and shown that our approach achieves a better accuracy than the state-of-the-art single hand tracker [27].

6 Acknowledgments

The authors acknowledge the help of Javier Romero and Jessica Purmort of MPI-IS regarding the acquisition of the personalized hand model, the assistance of Philipp Rybalov with annotation and the public software release of the FORTH tracker by the CVRL lab of FORTH-ICS, enabling comparison to [27]. Financial support was provided by the DFG Emmy Noether program (GA 1927/1-1).

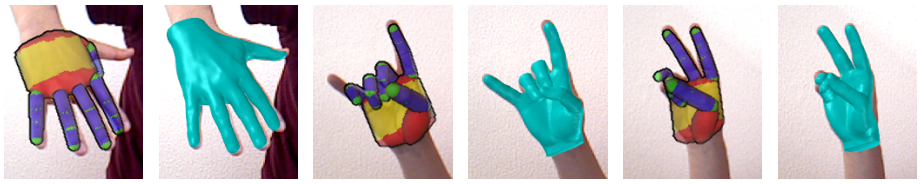

Fig. 2. Qualitative comparison with [27]. Each image pair corresponds to the pose estimate of the FORTH tracker (left) and our tracker (right)

References

1. Albrecht, I., Haber, J., Seidel, H.P.: Construction and animation of anatomically based human hand models. In: SCA. pp. 98–109 (2003)
2. Athitsos, V., Sclaroff, S.: Estimating 3d hand pose from a cluttered image. In: CVPR. pp. 432–439 (2003)
3. Ballan, L., Taneja, A., Gall, J., Van Gool, L., Pollefeys, M.: Motion capture of hands in action using discriminative salient points. In: ECCV. pp. 640–653 (2012)
4. Baran, I., Popović, J.: Automatic rigging and animation of 3d characters. TOG 26(3) (2007)
5. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. PAMI 24(4), 509–522 (2002)
6. Bregler, C., Malik, J., Pullen, K.: Twist based acquisition and tracking of animal and human kinematics. IJCV 56(3), 179–194 (2004)
7. de Campos, T., Murray, D.: Regression-based hand pose estimation from multiple cameras. In: CVPR. pp. 782–789 (2006)
8. Canny, J.: A computational approach to edge detection. PAMI 8(6), 679–698 (1986)
9. Chen, Y., Medioni, G.: Object modeling by registration of multiple range images. In: ICRA. pp. 2724–2729 (1991)
10. Ekvall, S., Kragic, D.: Grasp recognition for programming by demonstration. In: ICRA. pp. 748–753 (2005)
11. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, X.: Vision-based hand pose estimation: A review. CVIU 108(1-2), 52–73 (2007)
12. Felzenszwalb, P.F., Huttenlocher, D.P.: Distance transforms of sampled functions. Tech. rep., Cornell Computing and Information Science (2004)
13. Gall, J., Fossati, A., Van Gool, L.: Functional categorization of objects using real-time markerless motion capture. In: CVPR. pp. 1969–1976 (2011)
14. Gall, J., Yao, A., Razavi, N., Van Gool, L., Lempitsky, V.: Hough forests for object detection, tracking, and action recognition. PAMI 33(11), 2188–2202 (2011)
15. Hamer, H., Schindler, K., Koller-Meier, E., Van Gool, L.: Tracking a hand manipulating an object. In: ICCV. pp. 1475–1482 (2009)
16. Hamer, H., Gall, J., Weise, T., Van Gool, L.: An object-dependent hand pose prior from sparse training data. In: CVPR. pp. 671–678 (2010)
17. Heap, T., Hogg, D.: Towards 3d hand tracking using a deformable model. In: FG. pp. 140–145 (1996)
18. Holzer, S., Rusu, R., Dixon, M., Gedikli, S., Navab, N.: Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In: IROS. pp. 2684–2689 (2012)
19. Jones, M.J., Rehg, J.M.: Statistical color models with application to skin detection. IJCV 46(1), 81–96 (2002)
20. Keskin, C., Kra, F., Kara, Y., Akarun, L.: Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In: ECCV. pp. 852–863 (2012)
21. Kim, D., Hilliges, O., Izadi, S., Butler, A.D., Chen, J., Oikonomidis, I., Olivier, P.: Digits: freehand 3d interactions anywhere using a wrist-worn gloveless sensor. In: UIST. pp. 167–176 (2012)
22. Kyriazis, N., Argyros, A.: Physically plausible 3d scene tracking: The single actor hypothesis. In: CVPR. pp. 9–16 (2013)
23. Lewis, J.P., Cordner, M., Fong, N.: Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In: SIGGRAPH. pp. 165–172 (2000)

24. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface-quality hand tracking. In: ECCV. pp. 3–19 (2000)
25. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. CVIU 104(2), 90–126 (2006)
26. Murray, R.M., Sastry, S.S., Zexiang, L.: A Mathematical Introduction to Robotic Manipulation (1994)
27. Oikonomidis, I., Kyriazis, N., Argyros, A.: Efficient model-based 3d tracking of hand articulations using kinect. In: BMVC. pp. 101.1–101.11 (2011)
28. Oikonomidis, I., Kyriazis, N., Argyros, A.: Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In: ICCV. pp. 2088–2095 (2011)
29. Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Tracking the articulated motion of two strongly interacting hands. In: CVPR. pp. 1862–1869 (2012)
30. Paris, S., Durand, F.: A fast approximation of the bilateral filter using a signal processing approach. IJCV 81(1), 24–52 (2009)
31. Pons-Moll, G., Rosenhahn, B.: Model-Based Pose Estimation, pp. 139–170 (2011)
32. Rehg, J.M., Kanade, T.: Visual tracking of high dof articulated structures: an application to human hand tracking. In: ECCV. pp. 35–46 (1994)
33. Rehg, J., Kanade, T.: Model-based tracking of self-occluding articulated objects. In: ICCV. pp. 612–617 (1995)
34. Romero, J., Kjellström, H., Kragic, D.: Monocular real-time 3d articulated hand pose estimation. In: HUMANOIDS. pp. 87–92 (2009)
35. Romero, J., Kjellström, H., Kragic, D.: Hands in action: real-time 3d reconstruction of hands in interaction with objects. In: ICRA. pp. 458–463 (2010)
36. Rosales, R., Athitsos, V., Sigal, L., Sclaroff, S.: 3d hand pose reconstruction using specialized mappings. In: ICCV. pp. 378–387 (2001)
37. Rosenhahn, B., Brox, T., Weickert, J.: Three-dimensional shape knowledge for joint image segmentation and pose tracking. IJCV 73(3), 243–262 (2007)
38. Rusinkiewicz, S., Levoy, M.: Efficient variants of the icp algorithm. In: 3DIM. pp. 145–152 (2001)
39. Rusinkiewicz, S., Hall-Holt, O., Levoy, M.: Real-time 3d model acquisition. TOG 21(3), 438–446 (2002)
40. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: CVPR. pp. 1297–1304 (2011)
41. Sridhar, S., Oulasvirta, A., Theobalt, C.: Interactive markerless articulated hand motion tracking using rgb and depth data. In: ICCV. pp. 2456–2463 (2013)
42. Stenger, B., Mendonca, P., Cipolla, R.: Model-based 3D tracking of an articulated hand. In: CVPR. pp. 310–315 (2001)
43. Stolfi, J.: Oriented Proj. Geometry: A Framework for Geom. Computation (1991)
44. Teschner, M., Kimmerle, S., Heidelberger, B., Zachmann, G., Raghupathi, L., Fuhrmann, A., Cani, M.P., Faure, F., Magnetat-Thalmann, N., Strasser, W.: Collision detection for deformable objects. In: Eurographics. pp. 119–139 (2004)
45. Thayananthan, A., Stenger, B., Torr, P.H.S., Cipolla, R.: Shape context and chamfer matching in cluttered scenes. In: CVPR. pp. 127–133 (2003)
46. Tzionas, D., Gall, J.: A comparison of directional distances for hand pose estimation. In: GCPR. pp. 131–141 (2013)
47. Vaezi, M., Nekouie, M.A.: 3d human hand posture reconstruction using a single 2d image. IJHCI 1(4), 83–94 (2011)
48. Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. TOG 28(3), 63:1–63:8 (2009)