Motion Capture of Hands in Action using Discriminative Salient Points

Luca Ballan[1], Aparna Taneja[1], Juergen Gall[2], Luc Van Gool[1], and Marc Pollefeys[1]

[1] ETH Zurich, Switzerland
[2] MPI for Intelligent Systems, Germany

Abstract. Capturing the motion of two hands interacting with an object is a very challenging task due to the large number of degrees of freedom, self-occlusions, and similarity between the fingers, even in the case of multiple cameras observing the scene. In this paper we propose to use discriminatively learned salient points on the fingers and to estimate the finger-salient point associations simultaneously with the estimation of the hand pose. We introduce a differentiable objective function that also takes edges, optical flow and collisions into account. Our qualitative and quantitative evaluations show that the proposed approach achieves very accurate results for several challenging sequences containing hands and objects in action.

1 Introduction

Capturing the motion of hands is a very challenging computer vision problem that is also highly relevant for other areas like computer graphics, human-computer interaction and robotics. While intrusive methods like data gloves or color gloves [1] provide an acceptable solution for some applications, marker-less hand tracking is still an unsolved problem.

Although the problem has been studied since the nineties [2], most approaches have focused on tracking a single hand in isolation [3]. This, in itself, is very challenging due to the many degrees of freedom, as well as due to self-occlusions and similarity between the fingers. Despite these challenges, recent works [4–6] have addressed the problem of a hand interacting with an object. While an object may generate additional occlusions, these can be used to constrain the space of possible poses, as in [7, 6]. Both these approaches, however, assume that the manipulated object is rigid and not skin-colored, i.e., the object is relatively easy to track.

In this work, we address the even more challenging problem of capturing the articulated motion of two hands that interact with each other and with an additional object, as shown in Figure 1. In contrast to object manipulation by a single hand, two interacting hands cannot be separated based on color. Very recently, Oikonomidis et al. [8] have proposed an approach for tracking two interacting hands without objects. They employ a depth sensor to overcome the ambiguities of color images and use collision constraints, as in [6], to avoid hand intersections.

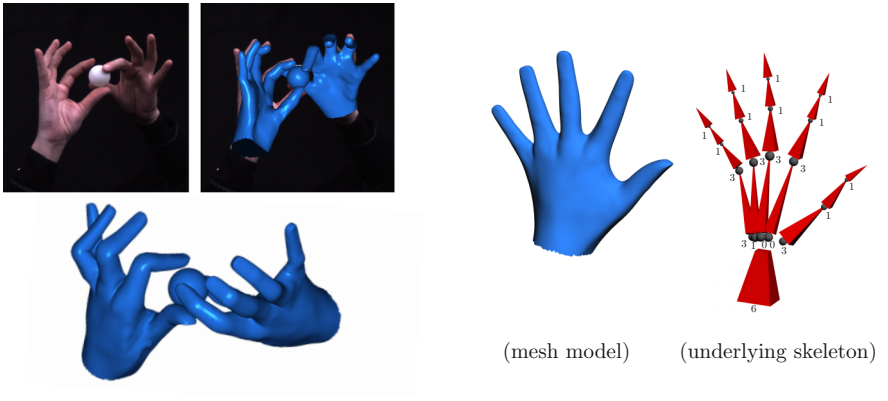

Fig. 1. (Left) Result obtained on a sequence where two hands were interacting with a ball. This sequence consists of a total of 73 degrees of freedom. (Right) Mesh model and underlying bone skeleton used to represent the hands in the scene. The numbers near each joint indicate the number of allowed degrees of freedom.

In this work, we propose a method that overcomes these ambiguities in a multi-camera setup. To this end, we make use of salient points, like finger tips, as in the earlier work of [2]. Differently from their scenario however, these salient points cannot be tracked continuously due to the huge amount of occlusions and the similarity in appearance of these features. Therefore we cannot rely on having a fixed association between the salient points and the respective fingers.

To cope with this, we propose a novel approach that solves the salient point association jointly with the hand pose estimation problem. In addition, we propose an almost everywhere differentiable objective function for pose estimation that takes into account edges, optical flow, salient points and collisions. In this way, we can resort to simple local optimization techniques without having to employ a sampling based optimization, as in [8].

In our experiments, we demonstrate that our approach can deal with very challenging sequences containing hands in action with and without an object, and it is also robust to errors in the salient point detections. In a quantitative comparison, we show that our iterative local optimization approach achieves significantly lower pose estimation errors than the sampling approach used in [8].

2 Related Work

In the survey [3], various methods for hand pose estimation have been reviewed in the context of human-computer interaction. Following their taxonomy, hand pose estimation approaches can be split into two categories: discriminative approaches that use classification or regression techniques directly on the image data, and generative approaches that use explicit hand models to recover the hand pose.

Generative approaches mainly differ within each other by the different visual cues extracted from the images, and by the optimization techniques employed

to solve the estimation problem. The most commonly used visual features are silhouettes and edges, but shading, color, and optical flow have also been used in the past [9, 10]. Depth has also been used [11, 12], and recently it has been revisited in the context of depth sensors [13].

In order to recover the hand pose based on these cues, several optimization techniques have been proposed. In one of the first approaches for 3D hand pose estimation [2], local optimization was adopted. Local optimization is still a very popular method for pose estimation due to its efficiency, however it requires a very carefully designed objective function in order to avoid local minima [10]. Other approaches rely on stochastic optimization techniques such as Kalman filter [14] and particle filter [15]. While particle filter and local optimization have been combined in [12] to improve the performance of these techniques in the high-dimensional space of hand poses, some approaches like [16] and [17] proposed to reduce the pose space using linear subspaces. As a drawback however, these methods can represent only a very limited number of hand poses. Other kinds of optimization techniques have also been explored, such as belief propagation [18, 4] and particle swarm optimization [19].

Discriminative approaches instead do not require an explicit hand model, but learn a mapping from image features to hand poses from a large set of training data [20–23]. Most of these methods process the frames independently, however temporal consistency can also be easily enforced [24, 5]. Although discriminative methods can recover from errors, their accuracy and the type of poses they can handle depend on the training data. Discriminative approaches are therefore not suitable for applications that require accurate hand pose estimation on a priori unknown actions. In this work, we propose a generative approach based on local optimization that uses a discriminatively trained salient point detector to address the problem of ambiguous poses in the objective functional.

As pointed out in the introduction, not much work has been done on hand motion capture in the context of interactions. [4] has considered hand tracking from depth data in the context of object manipulation. To assist this kind of tracking [7] proposed to learn a hand pose prior dependent on the type of object being manipulated, exploiting the idea that manipulating similar objects involves similar hand motions. In the context of object grasping, [5] built a database of $10,000$ hand poses with and without grasped objects to recover the hand pose from images using nearest neighbor search. Recently, [6] proposed to track the manipulated object and the hand at the same time to reduce the search space using collision constraints. In their work, objects were assumed to be simple enough to be modeled with shape primitives such as cylinders, ellipsoids or boxes.

While these works address the problem of capturing the motion of a single hand interacting with a rigid, non-skin colored object, we address the problem of capturing the motion of two hands interacting with each other and an additional object. There are only two very recent works [25, 8] that have addressed a comparable problem. In [8], particle swarm optimization (PSO) has been applied to the tracking of two interacting hands from depth data. It is an extension of [6] and

demonstrates the potential of PSO. However, the sampling technique adopted in this approach prevents very accurate pose estimates. Our experiments show that our method with local optimization significantly outperforms PSO in terms of accuracy. In [25], the motion of two interacting characters is captured. This approach however, relies on the assumption that both characters in the scene can be segmented in all the images based on their appearances and their shapes. This cannot be assumed in the case of interacting hands due to the similarity of their colors. Their tracking algorithm would indeed fail when the two hands touch each other, underlying the difficulty of the problem addressed in this work.

3 Scene Model

Each trackable element in the scene is modeled as a linear blend skinned model consisting of a surface mesh model and an underlying bone skeleton, as depicted in Figure 1. In this model, surface deformations are encoded using the linear blend skinning operator (LBS) [26], which maps skeleton configurations θ into mesh vertices positions $v_k(\theta)$. For a skeleton consisting of m bones, and $T_j(\theta)$ being the homogeneous transformation applied to the bone of index j, the position of the mesh vertex k is defined as

$$v_k(\theta) = \sum_{j=1}^{m} \alpha_{k,j} T_j(\theta) T_j(0)^{-1} v_k(0), \tag{1}$$

where the scalars $\alpha_{k,j}$ define how each bone j influences the position of each vertex k. $v_k(0)$ and $T_j(0)$ represent respectively the vertex positions and the bone transformations at configuration 0 (also known as the rigging configuration). $\alpha_{k,j}$, $v_k(0)$ and $T_j(0)$ are assumed to be known a priori.

In our scenario, both hands and objects are treated as a single combined linear blend skinned model. The skeleton underlying each hand consists of 20 bones, for a total of 35 degrees of freedom (DOF) per hand. Figure 1 shows how these DOF are distributed across all the joints. Note that the wrist is modeled with 6 DOF to allow for translation, while each knuckle is modeled with 3 DOF to account for small twists in the fingers. The hand mesh at the rigging pose $v_k(0)$ was acquired using multiview stereo and Poisson surface reconstruction on some images of the subject's hands. The skeleton was then fitted manually and a commercial modeling software was used to define the blending weights $\alpha_{k,j}$.

The global pose configuration θ is expressed in exponential coordinates (i.e., axis-angle representation followed by a translation) and is related to each bone transformation $T_j(\theta)$ by the $SE(3)$ exponential map operator, as in [27]. Bone angle constraints are imposed by forcing θ to belong to a space of allowed configurations Θ (linearly bounded). Constraints induced by collision and self-intersections are accounted for during the pose estimation.

4 Pose Estimation

Input videos are assumed to be synchronized and spatially calibrated. The recording cameras are assumed to be static for the entire action, and the projec-

Algorithm 1: Pose Estimation

θ_0 = pose estimated for the previous frame
$i = 0$, $\mathcal{C} = \emptyset$
Repeat until convergence
- Render mesh at pose θ_i
- Compute edge correspondences $\mathcal{E}$
- Compute optical flow correspondences $\mathcal{O}$
- Compute salient point correspondences $\mathcal{S}$
- $\mathcal{C} = \mathcal{C} \cup \{\text{collisions detected for pose } \theta_i\}$

$\theta_{i+1} = \arg\min_{\theta \in \Theta} \mathcal{F}^{\mathcal{E},\mathcal{O},\mathcal{S}}(\theta)$
$i = i + 1$

tion functions $\Pi_c : \mathbb{R}^3 \longrightarrow \mathbb{R}^2$, mapping 3D points into the image plane of each camera c, are assumed to be known.

The recorded videos are first preprocessed in order to extract the visual features necessary for the pose estimation. In this work, we focus on edges, optical flow and salient points. This information is then fed into our algorithm which aims at finding the model configuration θ which best explains the observed features. This is achieved by iteratively formulating and minimizing an objective functional accounting for all the acquired information.

Algorithm 1 shows an overview of the proposed approach. The algorithm concept is similar to the one proposed in [28] and [29]. At each iteration i, the current pose estimate θ_i is evaluated by generating and comparing compatible visual features with those extracted from the videos. This is performed by means of rendering first, and then, by generating a set of *vertex-to-image point correspondences* indicating how far each feature is from the corresponding feature on the actual images. Precisely, each correspondence is defined as a triplet (k, c, p), where k is a vertex index, c is a camera index, and p is a 2D point on the image plane of camera c. For a correct pose estimation, the vertex k should project onto the image point p in camera c, i.e., the configuration θ_i should be chosen in such a way that $\Pi_c(v_k(\theta_i)) = p$.

Given a list of edge correspondences $\mathcal{E}$, optical flow correspondences $\mathcal{O}$ and salient point correspondences $\mathcal{S}$, we look for the configuration θ which best satisfies all these observations by minimizing the following objective functional

$$\mathcal{F}^{\mathcal{E},\mathcal{O},\mathcal{S}}(\theta) = \sum_{(k,c,p) \in \mathcal{E} \cup \mathcal{O} \cup \mathcal{S}} \left\| \Pi_c(v_k(\theta)) - p \right\|^2 + \Gamma(\theta) \tag{2}$$

where $\Gamma(\theta)$ is a pose prior designed to softly penalize collisions and self-intersections. This term is described in detail in Section 4.2.

In order to generate valid edge correspondences $\mathcal{E}$, the edge maps of the mesh model at the current pose estimate θ_i are computed by thresholding the gradient of the rendered depth maps for each camera c. Each mesh vertex k lying on an edge of these maps, is matched with the closest edge pixel p in the corresponding input image of camera c, giving more preference to edges with

similar orientation. Optical flow correspondences $\mathcal{O}$ are computed as in [28], while salient point correspondences $\mathcal{S}$ are explained in the next section.

4.1 Salient points

In the case of close interactions between hands, classical features like edges and optical flow are insufficient to completely resolve for pose ambiguities. Image edges in fact, completely disappear when hands touch each other due to the similarities of their colors. To cope with these ambiguities, an additional stronger cue needs to be used.

To this end, we discriminatively learn the appearance of some characteristic features on the hands and detect them on each recorded video. In particular, we focused on finger nails because of their distinct appearance and their rigid deformation behavior. A Hough forest classifier [30] was trained on 400 manually labeled images of nails/non-nails seen from different viewing angles. The resulting classifier was then applied on each video sequence to generate a list of *salient image points*. On the mesh model side, the vertices corresponding to the center of each nail were marked as salient vertices. For a correct pose estimation, these salient vertices should always project onto salient image points on the videos.

Unfortunately, since the appearance of nails across fingers is very similar, training a classifier which also can discriminate between nails of different fingers is almost impossible. Exploiting temporal continuity by tracking the salient points in the videos can help, but in general, it can also generate ambiguous solutions when nails come close together or when they are completely occluded. We therefore need to live with this missing information and do not rely on having a fixed association between the detected salient image points and the salient vertices. This association needs to be performed jointly during the pose estimation and included as an additional unknown to our problem.

At each algorithm iteration i, the set of all the visible salient vertices on the mesh is computed for each camera c. Let $\Xi = \{\xi_1, \ldots, \xi_r\}$ denote the indices of such vertices for camera c, and let $\Delta = \{\delta_1, \ldots, \delta_d\}$ denote the set of all the salient image points detected by the classifier at the current frame on the same camera c. In order to find reasonable matches between salient vertices Ξ and salient image points Δ, we formulate a Bipartite Perfect Matching graph problem. A $|\Xi| \times |\Delta|$ matrix is defined with entries $w_{st} = \|\Pi_c\left(v_{\xi_s}\left(\theta_i\right)\right) - \delta_t\|$ encoding the distances between the projections of the salient vertices Ξ and their corresponding salient image points Δ.

Due to possible occlusions and inaccuracies in the current pose estimate θ_i, the resulting number of visible salient vertices might be different from the actual one. Moreover, false positives and missed detections resulting from the classifier might also hinder finding a perfect match between the two sets.

To cope with this, we first make the matrix $\{w_{st}\}_{st}$ square by adding virtual elements on both the sets and by setting their corresponding weights to infinity to ensure that they would never be chosen as candidate matches. Outliers in both Ξ and Δ are handled by adding two additional sets of virtual nodes, $\{\alpha_s\}_s$ and $\{\beta_t\}_t$, indicating whether the corresponding element ξ_s and δ_t is an outlier

or not. The matching problem is then formulated as an integer programming problem as follows

$$\begin{aligned} \arg\min \quad & \sum_{s,t} e_{st} w_{st} + \lambda \sum_{s} \alpha_s + \lambda \sum_{t} \beta_t & \\ \text{subject to} \quad & \sum_{s} e_{st} + \beta_t = 1 & \forall t \\ & \sum_{t} e_{st} + \alpha_s = 1 & \forall s \\ & e_{st}, \alpha_s, \beta_t \in \{0,1\} & \forall s,t \end{aligned} \tag{3}$$

where the binary variables $\{e_{st}\}_{st}$ indicate whether the salient image point δ_t is matched with the salient vertex ξ_s (in case of $e_{st} = 1$), or not (in case of $e_{st} = 0$). The parameter λ indicates the penalty cost for assigning an element to a virtual node α_s or β_t. In all our experiments this parameter was set to 60 pixels, so that salient vertices distant more than 60 pixels from all the salient image points are more prone to be recognized as outliers, and viceversa. The problem in Eq. 3 can be solved in polynomial time and the resulting $\{e_{st}\}_{st}$ are then used to generate a valid set of salient points correspondences $\mathcal{S}$.

4.2 Collision Term

It has been proven that taking collisions and self-intersections into account improves the pose estimation results [6] significantly. For this reason, we also account for such information in the proposed method.

Instead of relying on an additional model to explicitly account for collisions, like in [6], we chose a more generally applicable solution opting for a face-to-face collision detection approach on the original mesh. Collisions are then penalized using distance fields [31]. These fields are commonly used in graphics for computing distances between distinct scene elements, however they become computationally expensive in case of self-collisions. Because then a distinct distance field needs to be computed for each face in the model. To overcome such an issue, we chose to approximate these fields locally and to generate them only when needed, i.e., only when a collision is detected.

At each iteration i of the algorithm, a list of colliding mesh faces $\mathcal{C}$ is incrementally updated using the current pose estimate θ_i. This is quickly performed using Bounding Volume Hierarchies, as described in [32]. For each pair of colliding faces f_s and f_t, two local distance fields, $\Psi_{f_s \to f_t}$ and $\Psi_{f_t \to f_s}$, are defined for face f_s and face f_t respectively. The field $\Psi_{f_s \to f_t}$ is applied to the vertices of the colliding face f_t, and viceversa, the field $\Psi_{f_t \to f_s}$ is applied to the vertices of face f_s. The collision term $\Gamma(\theta)$ in Equation 2 is then defined as follows

$$\Gamma(\theta) = \sum_{(f_t, f_s) \in \mathcal{C}} \left(\sum_{k \in f_t} \Psi_{f_s \to f_t}(v_k(\theta)) + \sum_{k \in f_s} \Psi_{f_t \to f_s}(v_k(\theta)) \right). \tag{4}$$

The distance field $\Psi_{f_s \to f_t} : \mathbb{R}^3 \longrightarrow \mathbb{R}^+$ is defined in such a way that it penalizes all the vertices of face f_t which are inside the cone circumscribing

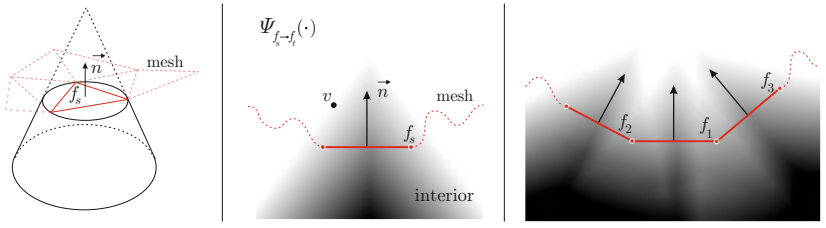

Fig. 2. (Left) Domain of the distance field $\Psi_{f_s \to f_t}$ generated by the face f_s. (Middle) Longitudinal section of the distance field $\Psi_{f_s \to f_t}$: darker areas correspond to higher penalties. (Right) Distance fields add up in case of multiple collisions.

face f_s in one of its circular sections, see Figure 2(left). The further a vertex v is inside this cone, the more it gets penalized, see Figure 2(middle). Figure 2(right) shows how multiple cones $\Psi_{f_s \to f_t}$ sum up in case of multiple collisions. For a mathematical formulation of $\Psi_{f_s \to f_t}$, please refer to the supplementary material. Since, the resulting $\Gamma(\theta)$ is differentiable almost everywhere, a local optimization on this functional will lead the collisions to resolve themselves in a few iterations.

4.3 Optimization

Due to the non-linear least squares formulation of the objective functional $\mathcal{F}$, the Levenberg-Marquard algorithm was used to accomplish a local optimization of $\mathcal{F}$ at each algorithm iteration i: using θ_i as an initial guess and constraining the solution to be inside Θ. The obtained solution θ_{i+1} is then used for the next iteration. This process is repeated iteratively until convergence.

By formulating $\mathcal{F}$ as a least squares problem, we inherently assume a Gaussian error on the observations. Therefore we need to explicitly account for outliers. This is done by computing for each bone, the mean and standard deviation of the motion that the bone would undergo if all the correspondences were considered. Correspondences suggesting a bone motion greater than twice the standard deviation of the estimated motion are excluded for the current optimization.

To avoid local minima that might result from the proposed optimization (due to the non-convexity of $\mathcal{F}$), we ensure that the final value of $\mathcal{F}(\theta)$ is below a predefined threshold, or else, a re-initialization using sampling (simulated annealing) is performed. In all our experiments, re-initialization was needed only twice.

5 Experimental Evaluation

The algorithm was tested on 7 real-world sequences with lengths varying from 300 to 950 frames, performing both hand to hand interactions and hand to object interactions. Videos were captured using a set up of eight synchronized

cameras recording FullHD footage at 50 fps. The recorded performances span a variety of actions, namely: praying, finger tips touching, fingers crossing, fingers crossing and twisting, fingers folding, fingers walking on the back of the hand, and holding and passing a ball. The total number of degrees of freedom used in all the sequences was 70. This number increases to 73 in the ball sequence.

Figure 5 shows one frame from each of the tested sequences and the obtained results overlayed with the original frames from two different viewing angles. Visual inspection reveals that the proposed algorithm works quite well even in challenging scenarios of very closely interacting hands with multiple occlusions. Results can be better appreciated in the supplementary videos available at http://cvg.ethz.ch/research/ih-mocap/.

5.1 Quantitative Evaluation and Comparison with Previous Work

In order to perform a quantitative evaluation of our algorithm, two experiments were conducted: one on a real sequence with manually marked groundtruth, and the other using synthetic data.

Synthetic data: We simulated two sequences: first, fingers crossing and folding, and second, holding and passing a ball, both similar to the ones captured in the real scenario. Videos were generated using a commercial rendering software. The pose estimation accuracy was then evaluated both in terms of error in the joints position, and in terms of error in the bones orientation.

Table 1 shows a quantitative evaluation of the algorithm performance with respect to the used visual features. It can be noted that, each feature contributes almost linearly to the accuracy of the algorithm, and also that the salient points $\mathcal{S}$ clearly boost its performance.

The results obtained using our iterative local optimization approach were also compared with those obtained using our implementation of HOPE [6]. For a fair comparison, we added the salient points term to the HOPE error functional and used the exact same parameters as in [6]. Specifically, we evaluated 64 particles over 40 generations. To account for the doubled number of DOF, we also tested HOPE with 128 particles. As expected HOPE works decently well in the tested sequences, however, due to its sampling nature it is bound to incur bigger errors. Our approach (LO) instead uses the local information of $\mathcal{F}$, i.e. the Jacobian, and drastically reduces the average error from $4.67mm$ to $1.49mm$.

In order to make these experiments as similar as possible to a real world scenario, we simulated noise in all the visual features. More precisely, edge detection errors were introduced by adding structural noise to the images, i.e. by adding and subtracting at random positions in each image, 100 circles of radius varying between 10 and 30 pixels. The optical flow features corresponding to those circles were also not considered. Errors in the salient point detector were simulated by randomly deleting detections as well as by randomly adding outliers in a radius of 200 pixels around the actual features. In the end, a gaussian noise of 5 pixels was introduced on the coordinates of the resulting salient points.

Figure 3(left) shows the influence of the salient point detector on the accuracy of the pose estimation in case of noisy data. This experiment was run with a

	Joints position error			Bones orientation error		
Used features	mean [mm]	std [mm]	max [mm]	mean [deg]	std [deg]	max [deg]
LO + $\mathcal{E}$	3.11	4.52	49.86	2.36	6.84	94.58
LO + $\mathcal{EC}$	2.50	2.89	52.94	1.98	4.57	91.89
LO + $\mathcal{ECO}$	2.38	2.25	16.84	1.84	3.81	60.09
LO + $\mathcal{ECOS}$	**1.49**	**1.44**	**13.27**	**1.88**	**3.90**	**44.51**
HOPE64 + $\mathcal{ECOS}$	4.86	3.69	31.05	4.35	7.11	58.61
HOPE128 + $\mathcal{ECOS}$	4.67	3.28	41.11	4.73	7.46	78.65

Table 1. Quantitative evaluation of the algorithm performance with respect to the used visual features: edges $\mathcal{E}$, collisions $\mathcal{C}$, optical flow $\mathcal{O}$, and salient points $\mathcal{S}$. LO stands for our iterative local optimization approach, while HOPE64 and HOPE128 stand for our implementation of [6] with 64 and 128 particles respectively, evaluated over 40 generations.

salient point false positive rate of 10%, and with varying detection rates. It is visible that the error quickly drops very close to its minimum even with a detection rate of only 30%.

Figure 3(right) shows the convergence rate of our algorithm for different number of iterations. It can be noted that the algorithm accuracy becomes quite reasonable after just 15 iterations. In our experiments, 15 iterations were used on an average in all the real world sequences.

Real sequence with manually marked data: We manually labeled some distinguishable points on the hands in all the videos of one real dataset. The corresponding points were then triangulated to obtain 3D points. The distance between these points and the corresponding vertices in the hand model was then calculated. This test was performed on the most challenging dataset we had, i.e., two hands interacting with a ball. Due to the large amount of manual effort involved, we only tracked 3 features throughout this sequence. Table 2 shows the tracking accuracy obtained in this experiment. Overall, the median of the tracking error is at maximum $1cm$. The reader should consider the fact that this error is also influenced by the inaccuracies introduced during the manual labeling. It can be noted that the finger with the lowest error (point 1) is actually the one interacting most closely with the ball in the majority of the sequence. A similar observation was made in [6].

Failure cases: Although the algorithm performs well in almost all the processed frames, pose estimation errors may appear in some very challenging cases due to severe occlusions. Figure 4 shows some such cases. Extending our approach with motion priors may help recover from such situations.

6 Conclusions

In this paper, we proposed a method to capture the articulated motion of two hands interacting with each other and with an object. To cope with the many

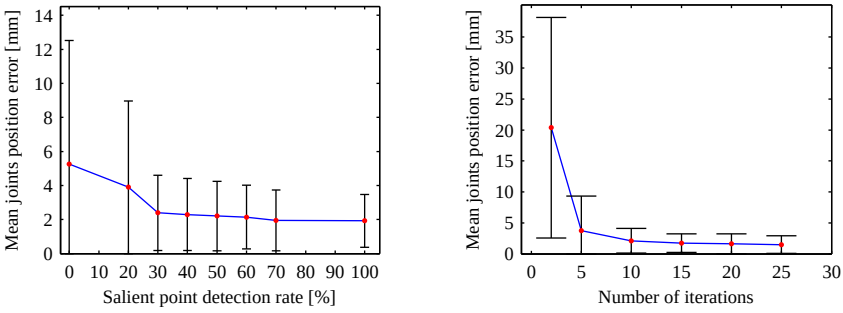

Fig. 3. Quantitative evaluation of the algorithm performance on noisy data, with respect to the salient point detection rate (left), and the number of iterations (right). Black bars indicate the standard deviation of the obtained error.

Points	median [mm]	mean [mm]	std [mm]	max [mm]
point 1	6.98	7.98	3.54	20.53
point 2	11.14	12.28	5.22	23.48
point 3	10.91	10.72	4.13	24.68

Table 2. Results obtained on the manually marked data. The table reports the distance in [mm] between the manually tracked 3D points and the corresponding vertices on the hand model. The figure shows the positions of the tracked points on the hand.

degrees of freedom involved, multiple occlusions and color/appearance similarity between the hands and the fingers, we chose to use multiple visual features such as edges, optical flow, salient points, and collisions to estimate the articulated pose within a single differentiable function.

We observed that the usage of discriminatively learnt salient points drastically improve the pose estimation accuracy, particularly in case of strong occlusions and close interaction between fingers, i.e., in the cases where features like edges and optical flow may not be reliable. To overcome the ambiguities of assigning salient points to the corresponding fingers, we proposed to solve this association simultaneously with the estimation of the hand pose. To handle self-intersection and collisions between elements in the scene, we proposed to use distance fields and to generate them locally only when needed.

While our experiments showed that the proposed method, with local optimization, achieves a significantly lower error than a state-of-the-art hand tracking method based on an evolutionary algorithm, a combination of both optimization approaches might be interesting to investigate. Experiments in real world scenarios demonstrate that our approach can deal with very challenging sequences containing hands in action with and without an object and it is also

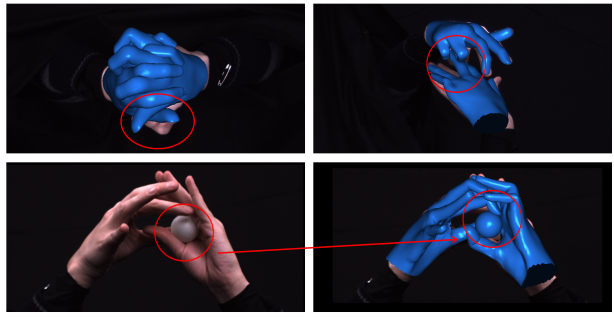

Fig. 4. Examples of cases where the algorithm exhibits pose estimation errors due to severe occlusions.

robust to errors in the salient point detections. We believe that the concept of solving for salient point associations and pose estimation together will also be relevant for other tracking applications.

7 Acknowledgements

The research leading to these results has received funding from the ERC grant #210806 4DVideo under the EC's 7th Framework Programme (FP7/2007-2013), and from the Swiss National Science Foundation.

References

1. Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. ACM Transaction on Graphics **28** (2009)
2. Rehg, J.M., Kanade, T.: Visual tracking of high dof articulated structures: an application to human hand tracking. In: ECCV. (1994) 35–46
3. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, X.: Vision-based hand pose estimation: A review. CVIU **108** (2007) 52–73
4. Hamer, H., Schindler, K., Koller-Meier, E., Van Gool, L.: Tracking a hand manipulating an object. In: ICCV. (2009) 1475–1482
5. Romero, J., Kjellström, H., Kragic, D.: Hands in action: real-time 3d reconstruction of hands in interaction with objects. In: ICRA. (2010) 458–463
6. Oikonomidis, I., Kyriazis, N., Argyros, A.: Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In: ICCV. (2011)
7. Hamer, H., Gall, J., Weise, T., Van Gool, L.: An object-dependent hand pose prior from sparse training data. In: CVPR. (2010) 671–678
8. Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Tracking the articulated motion of two strongly interacting hands. In: CVPR. (2012)
9. Lu, S., Metaxas, D., Samaras, D., Oliensis, J.: Using multiple cues for hand tracking and model refinement. In: CVPR. (2003)

10. de La Gorce, M., Fleet, D.J., Paragios, N.: Model-based 3d hand pose estimation from monocular video. PAMI **33** (2011) 1793–1805
11. Delamarre, Q., Faugeras, O.D.: 3d articulated models and multiview tracking with physical forces. CVIU **81** (2001) 328–357
12. Bray, M., Koller-Meier, E., Van Gool, L.: Smart particle filtering for high-dimensional tracking. CVIU **106** (2007) 116–129
13. Oikonomidis, I., Kyriazis, N., Argyros, A.: Efficient model-based 3d tracking of hand articulations using kinect. In: BMVC. (2011)
14. Stenger, B., Mendonca, P., Cipolla, R.: Model-based 3D tracking of an articulated hand. In: CVPR. (2001) 310–315
15. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface-quality hand tracking. In: ECCV. (2000) 3–19
16. Heap, T., Hogg, D.: Towards 3d hand tracking using a deformable model. In: International Conference on Automatic Face and Gesture Recognition. (1996)
17. Wu, Y., Lin, J., Huang, T.: Capturing natural hand articulation. In: ICCV. (2001) 426–432
18. Sudderth, E., Mandel, M., Freeman, W., Willsky, A.: Visual Hand Tracking Using Nonparametric Belief Propagation. In: Workshop on Generative Model Based Vision. (2004) 189–189
19. Oikonomidis, I., Kyriazis, N., Argyros, A.: Markerless and efficient 26-dof hand pose recovery. In: ACCV. (2010) 744–757
20. Rosales, R., Athitsos, V., Sigal, L., Sclaroff, S.: 3d hand pose reconstruction using specialized mappings. In: ICCV. (2001) 378–387
21. Athitsos, V., Sclaroff, S.: Estimating 3d hand pose from a cluttered image. In: CVPR. (2003) 432–439
22. de Campos, T., Murray, D.: Regression-based hand pose estimation from multiple cameras. In: CVPR. (2006) 782–789
23. Salzmann, M., Urtasun, R.: Combining discriminative and generative methods for 3d deformable surface and articulated pose reconstruction. In: CVPR. (2010)
24. Stenger, B., Thayananthan, A., Torr, P.: Model-based hand tracking using a hierarchical bayesian filter. PAMI **28** (2006) 1372–1384
25. Liu, Y., Stoll, C., Gall, J., Seidel, H.P., Theobalt, C.: Markerless motion capture of interacting characters using multi-view image segmentation. In: CVPR. (2011) 1249–1256
26. Lewis, J.P., Cordner, M., Fong, N.: Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. SIGGRAPH (2000) 165–172
27. Bregler, C., Malik, J., Pullen, K.: Twist based acquisition and tracking of animal and human kinematics. IJCV **56** (2004) 179–194
28. Ballan, L., Cortelazzo, G.M.: Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. In: 3DPVT. (2008)
29. Brox, T., Rosenhahn, B., Gall, J., Cremers, D.: Combined region- and motion-based 3d tracking of rigid and articulated objects. PAMI **32** (2010) 402–415
30. Gall, J., Yao, A., Razavi, N., Van Gool, L., Lempitsky, V.: Hough forests for object detection, tracking, and action recognition. PAMI **33** (2011) 2188–2202
31. Jones, M.W., Baerentzen, J.A., Sramek, M.: 3d distance fields: A survey of techniques and applications. IEEE Transactions on Visualization and Computer Graphics **12** (2006) 581–599
32. 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. (2004) 119–139

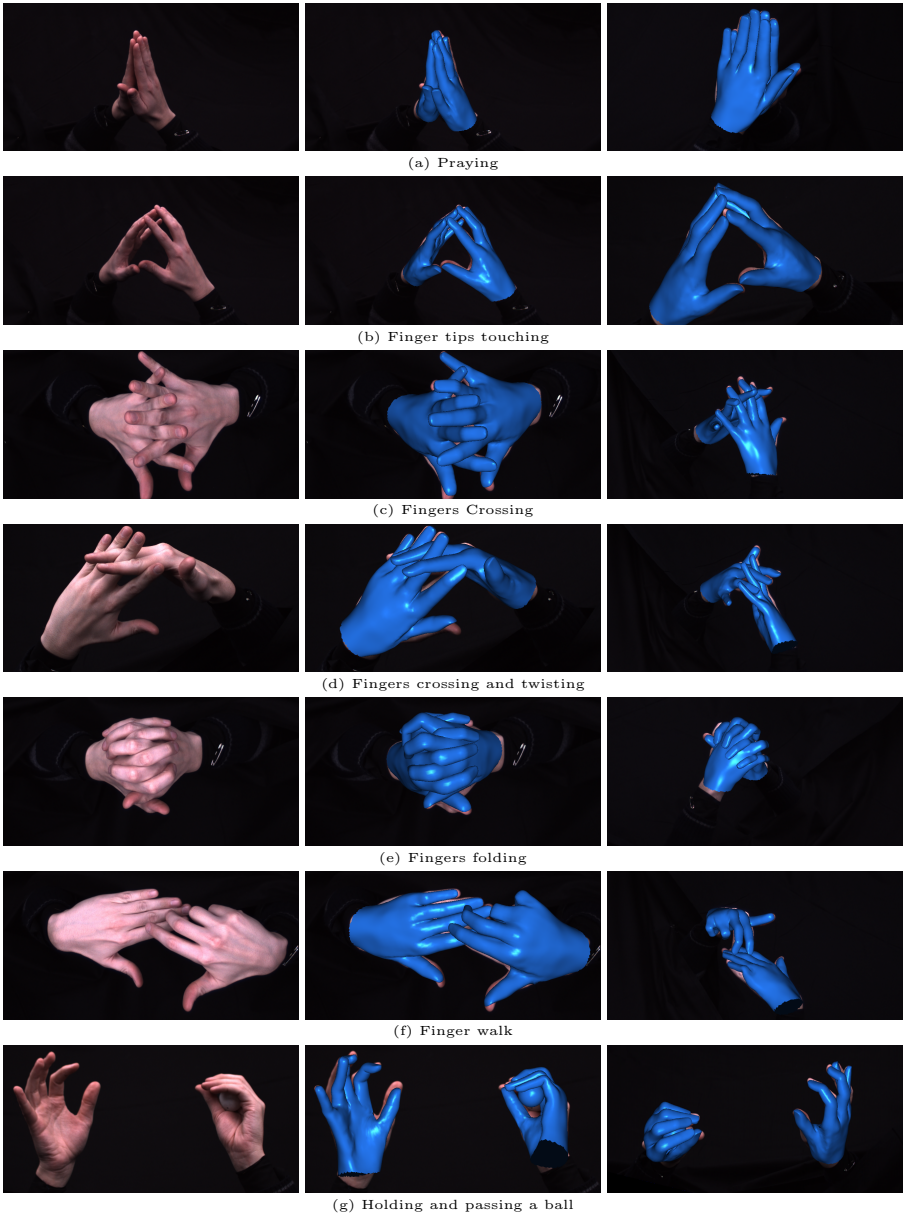

Fig. 5. Some of the obtained results. (Left) Input images. (Center) Obtained results overlayed with the images in the left column. (Right) Obtained results from another viewpoint. (Images have been zoomed in for better visualization of the results.)