2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Real-Time Hand Tracking using Synergistic Inverse Kinematics

Matthias Schröder[1], Jonathan Maycock[2], Helge Ritter[2], Mario Botsch[1]
[1]Computer Graphics & Geometry Processing Group, [2]Neuroinformatics Group,
Bielefeld University, Germany

***Abstract*— We present a method for real-time bare hand tracking that utilizes natural hand synergies to reduce the complexity and improve the plausibility of the hand posture estimation. The hand pose and posture are estimated by fitting a virtual hand model to the 3D point cloud obtained from a Kinect camera using an inverse kinematics approach. We use real human hand movements captured with a Vicon motion tracking system as the ground truth for deriving natural hand synergies based on principal component analysis. These synergies are integrated in the tracking scheme by optimizing the posture in a reduced parameter space. Tracking in this reduced space combined with joint limit avoidance constrains the posture estimation to natural hand articulations. The information loss associated with dimension reduction can be dealt with by employing a hierarchical optimization scheme. We show that our synergistic hand tracking approach improves runtime performance and increases the quality of the posture estimation.**

I. Introduction

Tracking the complete articulation of a freely moving hand is a problem that is an ongoing research topic and has numerous applications in robotics. Many existing hand tracking solutions either require the user to wear cumbersome equipment, are expensive or are inadequate for real-time tracking. Some methods that use consumer-level depth sensors can detect the positions of individual fingers and provide a means for rough gesture interaction, but do not accurately reconstruct the user's hand posture with full degrees of freedom (DoFs).

We have built a hand tracking system that uses a Kinect camera to estimate the full articulation of a user's bare hand in real-time. Our method is a generative approach that is based on fitting a virtual hand model to the 3D point cloud obtained from the Kinect sensor's depth camera. We estimate the hand articulation by finding the pose and posture parameters that minimize the error between the observed point cloud and the model surface using inverse kinematics. In doing so, we find the deformation of the 3D hand model that best approximates the observed state of the user's hand.

A prevalent issue in tracking a highly articulated object like a hand is the number of DoFs that must be optimized. The analysis of hand synergies aims to identify high-level relationships in hand articulation in order to sensibly reduce the dimensionality of hand posture representations. We obtain such hand synergies through the principal component analysis of motion capture data and use them directly in the tracking process to reduce the parameter space and to naturally constrain the hand posture estimation.

This paper extends our recent workshop paper [1] in several aspects: First, we speed up the inverse kinematics with a linear prediction step and a cylinder model; then, we employ joint limit avoidance to ensure physically plausible hand postures. We also propose a highly robust hierarchical optimization scheme. Finally we analyze the performance of our hand tracking system with a detailed evaluation.

Fig. 1. The user's hand posture is estimated in real-time by fitting a 3D hand model to the Kinect point cloud using inverse kinematics.

II. Related Work

There are two main approaches to the hand pose estimation problem, namely appearance based [2], [3] and model based methods [4], [5]. In a recent paper [6], we used a data-driven appearance based approach to control an anthropomorphic robot hand. A color glove was detected and using a nearest neighbor search in an image database the closest matching image and corresponding posture and coarse rotation were retrieved. However, appearance-based methods suffer when the hand strays from configurations that are not known and therefore can perform poorly in certain free moving hand situations. It is for this reason that we have decided to investigate a model based approach.

Recently, existing model based approaches that heretofore had proved too computationally expensive for real-time applications [7] are now becoming feasible. Oikonomidis *et al.* presented impressive results using a multi-camera setup [4] and using the Kinect camera [8], [9] with variants of particle swarm optimization, but these approaches suffer from high computational complexity and had to be optimized to run on a GPU. Ballan *et al.* [5] used features such as edges, optical flow, and salient points extracted from videos in a multi-camera setup to estimate the articulated pose of hands interacting with objects within a single objective function. They achieved lower posture estimation errors than those of Oikonomidis *et al.* [4], but thus far their approach is not real-time. Wang *et al.* [10] realized motion capture of hand grasping and manipulation data by simultaneously modeling hand articulation, object movement, and interactions between the two in an optimization framework. They obtained physi-

978-1-4799-3684-7/14/$31.00 ©2014 IEEE

cally accurate results, but their method is also not real-time. The so-called *curse of dimensionality* is an issue that has to be addressed by all hand posture estimation approaches and a possible solution we have considered is to use synergies to reduce the associated computational complexity.

Bernstein [11] was the first to come up with the idea of synergies and defined them to be high-level control schemes for kinematic parameters. He suggested that they could provide a mechanism by which the central nervous system controls the high DoF human hand in an efficient way. It was not, however, until Santello *et al.* [12] published their paper on hand synergies that the rehabilitative protheses and robots research communities realized their potential in terms of controlling articulated hands and arms. Santello revealed that 90% of the variance in the data of grasps directed towards household objects could be described by as little as 3 principal components (PCs). Many other studies have since supported this view [13], [14].

Using synergies or other methods to reduce the dimensionality of the problem of hand pose estimation for hand tracking applications has a precedence. In an early paper, Lee and Kunii [15] placed a set of constraints on joint angle limits and movement types to reduce the DoFs or reject infeasible inverse kinematics solutions. Wu *et al.* [16] used the fact that hand articulations could be represented in a lower dimensional space to perform a Monte Carlo tracking algorithm. Unlike Lee and Kunii, they were able to track the hand in real-time, but their method was view dependent and rotation and scaling were not considered. Another view dependent hand tracking particle filter approach [17] also reduced the dimensionality of the problem by using independent component analysis to compute basis components for all fingers. Bray *et al.* [18] used smart particle filtering to efficiently explore the high-dimensional search space with fewer samples.

We use dimension reduction in order to simplify and constrain the problem of inverse kinematics. A similar concept was previously employed in the work of Grochow *et al.* [19], who presented an inverse kinematics system based on a learned model of human poses. Safonova *et al.* [20] also used dimension reduction to synthesize realistic human motion. Using a synergistic approach to reduce the DoFs of a virtual hand model, we also reduce the high computational complexity associated with model based approaches, while at the same time realizing realistic (view unrestricted) real-time bare hand tracking.

III. Kinematic Hand Model

We use a kinematic hand model consisting of 16 joints: three for the proximal, intermediate and distal phalanges of each finger and one wrist joint. The articulation of the hand is represented by 20 degrees of freedom in our model: each finger joint has a flexion-extension axis and the fingers' base joints each have an additional abduction-adduction axis. In addition to the 20 joint angles controlling the hand's posture, the pose of the hand is represented by 6 degrees of freedom for the global translation and rotation. In total we use 26 parameters to control the pose and posture of the hand in our system.

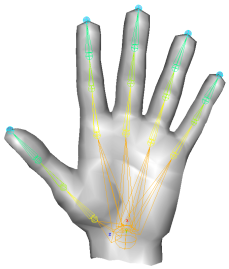


Fig. 2. Virtual hand model with its control skeleton.

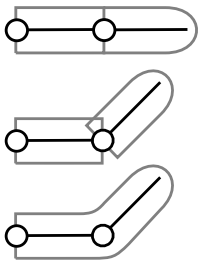


Fig. 3. Rest pose, forward kinematics and skinning.

These parameters and the kinematic chains of the joint hierarchy define the forward kinematics of the hand, which can be expressed in terms of a product of affine transformations. A joint's *local* transformation consists of the rotation defined by its joint angle parameters and the translation relative to its parent joint, if there is one. The *global* transformation matrix $\mathbf{T}_j$ of joint j is given by the product of the local transformations along its kinematic chain:

$$\mathbf{T}_j = \prod_{i=1}^{n} T_i(\theta_i), \tag{1}$$

where $T_i(\theta_i)$ is the local transformation matrix associated with the element θ_i of the kinematic parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{26})^T$.

The virtual hand used to approximate the user's hand posture in our system is represented as a triangle mesh and is deformed according to the articulation of the joints defined in the kinematic hand model. The joint hierarchy serves as the *skeleton* of the virtual hand model, as depicted in Figure 2. A point $\mathbf{v}$ on the mesh surface can be transformed relative to a joint j based on the forward kinematics of the skeleton:

$$\mathbf{v}' = \mathbf{T}_j \hat{\mathbf{T}}_j^{-1} \mathbf{v}, \tag{2}$$

where $\hat{\mathbf{T}}_j$ is the rest pose transformation of joint j and its inverse is used to transform $\mathbf{v}$ to the joint's local coordinate frame. Since the transformation matrices $\mathbf{T}_j$ depend on the parameter vector $\boldsymbol{\theta}$, the transformation of a point $\mathbf{v}$ based on the skeleton can be expressed as a function of the parameters: $\mathbf{v} = \mathbf{v}(\boldsymbol{\theta})$. We use this expression to calculate the forward kinematics during the tracking process.

However, computing the deformed hand model in this manner results in a blocky, piecewise rigid articulation. We therefore use a different animation technique for *visualization* than for *tracking*. For visualization we obtain a smooth deformation using linear blend skinning [21], which computes deformed vertex positions by linearly blending the transformation matrices of the joints influencing a vertex, which is specified by a set of convex weights $(\omega_1, \ldots, \omega_{16})$ for each vertex. This results in a smooth deformation of the virtual hand model in accordance to the control parameters of the kinematic model. Figure 3 illustrates the forward kinematics and skinning for a kinematic chain of two joints.

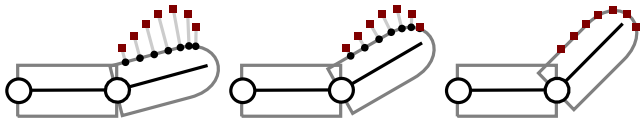


Fig. 4. Inverse kinematics posture estimation. The red squares are target positions in the sensor point cloud, the black circles are the corresponding effector positions on the surface of the model. The joint angle is updated such that the target-effector error is minimized.

IV. Inverse Kinematics Hand Tracking

In our hand tracking approach, the pose and posture of the user's hand are estimated by fitting the virtual hand model to the point cloud obtained from a Microsoft Kinect sensor. By finding the articulation of the hand model that minimizes the distance to the point cloud, the state of the user's hand that causes the observation is approximated.

The point cloud is calculated from the Kinect's color and depth images based on a precomputed RGBD-mapping, which maps color values to the pixel coordinates of the depth image and uses the camera parameters of the Kinect's color and depth cameras to calculate the global 3D positions of the sensor points. The hand is then segmented by detecting skin-colored pixels and omitting points whose coordinates are outside of a predefined working volume. The remaining points define the target constraints for the hand model fitting.

These target points are matched to their spatially closest points on the surface of the hand model. Based on these point-to-point correspondences we formulate the problem of estimating the posture of the hand as finding the posture parameters (joint angles and global pose) that transform the hand's skeleton such that the error between the deformed model and target positions is minimized. This is an inverse kinematics (IK) problem in which the points on the model surface $\mathbf{s}_i$ are regarded as effector positions relative to the skeleton, which are constrained to move towards their corresponding target positions $\mathbf{t}_i$ in the sensor point cloud. Figure 4 illustrates the principle with a simplified example.

A. Inverse Kinematics

We denote the effector positions by a stacked vector $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_k)^T$ and the target positions by $\mathbf{t} = (\mathbf{t}_1, \ldots, \mathbf{t}_k)^T$. As stated in Section III, the effector positions can be expressed as functions of the parameters $\mathbf{s}_i = \mathbf{s}_i(\boldsymbol{\theta})$ or $\mathbf{s} = \mathbf{s}(\boldsymbol{\theta})$. The IK problem, $\mathbf{t} = \mathbf{s}(\boldsymbol{\theta})$, can then be solved by iteratively finding a parameter update $\Delta\boldsymbol{\theta}$ from the previous frame $\boldsymbol{\theta}$ that minimizes the objective function:

$$E(\Delta\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{s}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) - \mathbf{t}\|^2 + \frac{1}{2}\|\mathbf{D}\Delta\boldsymbol{\theta}\|^2 \qquad (3)$$

The first term models the least squares error between the effectors $\mathbf{s}_i$ and their target positions $\mathbf{t}_i$. The second term regularizes the underdetermined problem by damping the parameter update $\Delta\boldsymbol{\theta}$ with a diagonal matrix $\mathbf{D}$ of damping weights $\lambda_1, \ldots, \lambda_{26}$. In our system we use no damping ($\lambda_i = 0$) for the six pose parameters and only a small amount of damping ($\lambda_i = 1.0$) for the 20 posture parameters.

We minimize (3) using a Gauss-Newton approach, each iteration of which involves solving the linear system

$$\left(\mathbf{J}^T\mathbf{J} + \mathbf{D}\right)\,\delta\boldsymbol{\theta} = \mathbf{J}^T\mathbf{e}, \qquad (4)$$

where $\mathbf{e} = \mathbf{t} - \mathbf{s}$ is the current target-effector error and $\mathbf{J}$ is the $(3k \times 26)$ Jacobian matrix of the effector positions

$$\mathbf{J} = \frac{\partial \mathbf{s}}{\partial \boldsymbol{\theta}} = \left(\frac{\partial \mathbf{s}_i}{\partial \theta_j}\right)_{i,j}. \qquad (5)$$

The straightforward calculation of $\mathbf{J}$ is described in [22]. Solving the system (4) yields the update direction $\delta\boldsymbol{\theta}$, whose step size typically is determined by a line search. We start with $\delta\boldsymbol{\theta}$ and successively halve it ($\delta\boldsymbol{\theta} \leftarrow \frac{1}{2}\delta\boldsymbol{\theta}$) until the error eventually decreases, i.e., $E(\Delta\boldsymbol{\theta} + \delta\boldsymbol{\theta}) < E(\Delta\boldsymbol{\theta})$. Only then the update $\Delta\boldsymbol{\theta} \leftarrow \Delta\boldsymbol{\theta} + \delta\boldsymbol{\theta}$ is accepted. This process is iterated until the Gauss-Newton minimization converges, which typically requires 5–10 iterations. As a starting value for $\Delta\boldsymbol{\theta}$ we simply use the update from the previous frame, i.e., we use a linear prediction as initial guess.

While our technique is very similar to the popular damped least squares method described in [22], it differs in two important aspects: The selective damping (cf. [23]) by $\mathbf{D}$ (instead of $\lambda\mathbf{I}$) increases responsiveness and performance of the system and the step size control for $\delta\boldsymbol{\theta}$ improves robustness by preventing oscillations. Without this step size control, a much higher damping is required to keep the tracking stable, which however leads to a significantly higher "laggyness".

B. Joint Limit Avoidance

In order to prevent the IK optimization from generating physically impossible hand postures, we constrain the posture parameters to plausible value ranges by adapting the joint limit avoidance of Chan and Dubey [24] to our setup. A posture parameter θ_i is slowed down by increasing its damping parameter λ_i in the matrix $\mathbf{D}$ as soon as it reaches its lower or upper joint limit $\theta_{i,\min}$ or $\theta_{i,\max}$, respectively. To this end a joint limit function H_i is defined as

$$H_i(\theta_i) = \frac{(\theta_{i,\max} - \theta_{i,\min})^2}{(\theta_{i,\max} - \theta_i)\,(\theta_i - \theta_{i,\min})},$$

and the damping parameter λ_i is scaled by $(1 + |\partial H_i/\partial\theta_i|)$ iff θ_i is moving *towards* its limits. The latter criteria is characterized by an increase of $|\partial H_i/\partial\theta_i|$ from the previous frame to the current, i.e., if $\Delta\,|\partial H_i/\partial\theta_i| \geq 0$. If the parameter is moving away from its limit ($\Delta\,|\partial H_i/\partial\theta_i| < 0$), its movement is unrestricted. This simple method causes parameters that move close towards their limits to slow down and to virtually stop when the joint limit is nearly reached. We obtain the joint limits $\theta_{i,min}$ and $\theta_{i,max}$ by extracting the minimum and maximum values from a database containing real human hand postures (see Section V).

C. Hand Tracking Process

The overall hand tracking process is illustrated in Figure 5. After segmenting the hand in the Kinect point cloud and finding the target-effector correspondences, the pose estimation is initialized by finding the rigid transformation between the

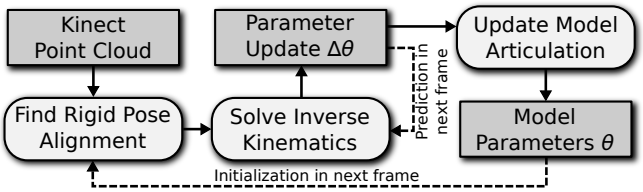


Fig. 5. Schematic overview of the overall hand tracking process.

target and effector point clouds using the well known rigid iterative closest points (ICP) approach of Horn [25], using the posture parameters from the previous frame. In the first frame, this ICP fitting provides a good initialization if the posture and orientation of the user's hand roughly matches the initial neutral state of the hand model.

Then the correspondences are updated and used as input for the iterative IK-based pose and posture estimation. During this process, the parameter update is computed according to (4) and the effector points are moved according to the updated skeleton forward kinematics. This process is iterated until the target-effector error converges. In practice, our IK optimization usually takes less than 10 Gauss-Newton iterations and for real-time tracking 5 iterations are sufficient.

The process of recomputing the correspondences and solving the IK optimization is iterated several times in a non-rigid ICP manner. As a result the virtual hand model is aligned with the observation point cloud, which yields the hand pose and posture estimation.

V. Principal Component Analysis

In the approach outlined in the previous section all 26 parameters of the kinematic model are freely optimized within their joint limits during the IK update. While this allows for high freedom of movement and mostly yields plausible hand articulations, it can also result in inaccurate or unnatural hand posture reconstructions in cases of incomplete or ambiguous sensor data.

To overcome this problem, the space of possible hand postures can be reduced in a meaningful way using hand synergies derived from the principal component analysis of a training data set containing real human hand motions. Performing dimension reduction based on the most significant principal components provides a way to decrease the number of parameters that need to be optimized and to implicitly constrain the estimated hand postures to plausible ones resulting from the training data.

In order to obtain a suitable training data set, we captured various human hand motions using a Vicon motion tracking system [26]. The positions of 16 retro-reflective markers placed on a human hand were tracked by the Vicon system and afterwards used to calculate joint angles corresponding to our kinematic model [27]. The global pose information was removed from the data, since they have no influence on the hand posture synergies we want to exploit. The recorded hand motions include various manual interaction tasks, such as different grasping and twisting motions, sign language, and general hand movements exploring the hand's natural degrees of freedom. This gave us a varied set of hand postures that covered many aspects of natural hand articulation.

A. Principal Component Space

The final data matrix $\mathbf{X}$ of m entries of the 20-dimensional posture data was pre-processed to have zero mean before PCA was performed on it. This resulted in a 20×20 matrix $\mathbf{V}$ of eigenvectors and the set of 20 eigenvalues $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{20})$. Taking the eigenvectors in $\mathbf{V}$ corresponding to the l largest eigenvalues yields a $20 \times l$ matrix of principal components, $\mathbf{C}$. Since the data used for PCA only contains the 20 joint angles and not the additional 6 pose DoFs, we construct the conversion matrix $\mathbf{M}$ that maps from the reduced $(6 + l)$-dimensional principal component-space (PC-space) to the $(6 + 20)$-dimensional parameter space:

$$\mathbf{M} = \begin{pmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{C} \end{pmatrix}, \tag{6}$$

where $\mathbf{I}$ is a 6×6 identity matrix, requiring the global pose parameters to be the first 6 in the parameter vector. The full parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{6+20}$ can be converted to the reduced parameter vector in PC-space, $\boldsymbol{\alpha} \in \mathbb{R}^{6+l}$, by the mapping (and vice versa by its inverse)

$$\boldsymbol{\alpha} = \mathbf{M}^T(\boldsymbol{\theta} - \boldsymbol{\mu}), \quad \text{and} \quad \boldsymbol{\theta} = \mathbf{M}\boldsymbol{\alpha} + \boldsymbol{\mu}. \tag{7}$$

Here, $\boldsymbol{\mu} \in \mathbb{R}^{26}$ is the mean of the data matrix $\mathbf{X}$ with six leading zero-entries for the pose DoFs. This allows the PC-space parameters to be expressed as a function of the kinematic parameters, $\boldsymbol{\alpha} = \boldsymbol{\alpha}(\boldsymbol{\theta})$, and vice versa, $\boldsymbol{\theta} = \boldsymbol{\theta}(\boldsymbol{\alpha})$. In order to perform inverse kinematics hand tracking (described in Section IV) using the reduced PC-space parameters, the parameter update rule must be adapted.

Given the above mapping, the forward kinematics of an effector point $\mathbf{s}_i$ can be written as a function of the PC-space parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_l)^T$: $\mathbf{s}_i = \mathbf{s}_i(\boldsymbol{\alpha}) = \mathbf{s}_i(\boldsymbol{\theta}(\boldsymbol{\alpha}))$. According to the chain rule, the $3k \times l$ Jacobian matrix $\mathbf{J}_{PC}$ of the effector positions w.r.t. the PC-parameters $\boldsymbol{\alpha}$ is:

$$\mathbf{J}_{PC} = \frac{\partial \mathbf{s}}{\partial \boldsymbol{\alpha}} = \frac{\partial \mathbf{s}}{\partial \boldsymbol{\theta}} \cdot \frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{\alpha}} = \mathbf{J} \cdot \mathbf{M}, \tag{8}$$

where $\mathbf{J}$ is the Jacobian matrix defined in (5). The joint limit avoidance described in Section IV-B can be applied to the PC-parameters in exactly the same way if upper and lower limits $\alpha_{i,\min}$ and $\alpha_{i,\max}$ for the PC-space parameters α_i are available. We determine these parameter limits by iterating through all postures in the training data set, projecting them into PC-space, and storing the minimum and maximum values along all PC-axes. The PC-space parameter update $\Delta\boldsymbol{\alpha}$ is obtained by replacing $\mathbf{J}$ by $\mathbf{J}_{PC}$ and $\mathbf{D}$ by an analogous $(6 + l) \times (6 + l)$ damping matrix in (4).

This facilitates hand tracking as described in Section IV in the reduced PC-space, which decreases the size of the matrices in the update calculation and reasonably constrains the possible hand postures estimated by the tracking system. The number l of PCs used for the dimension reduction depends on the distribution of the data variance among the principal directions (see Section VI).

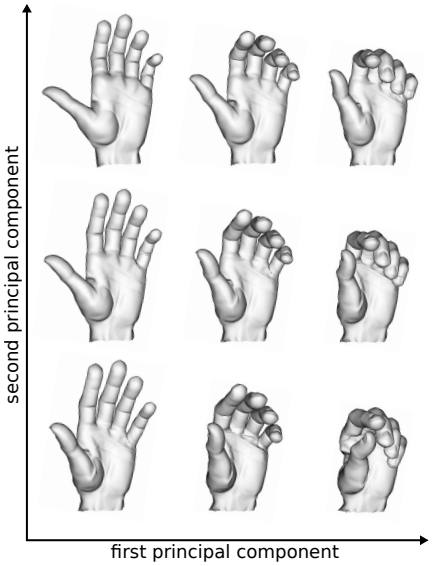


Fig. 6. Visualization of the DoFs represented by the first and second most significant principal components of the captured hand posture data set containing grasp movements.

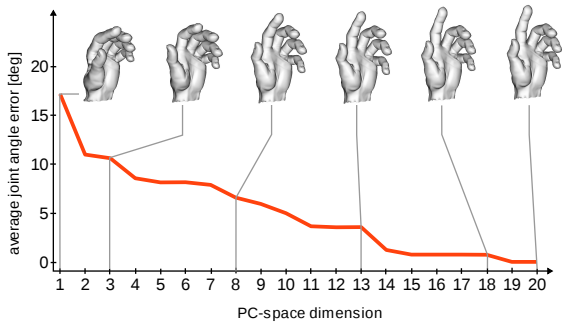


Fig. 7. Posture reconstruction and approximation error for one particular posture with increasing number of principal components. The hand postures on top show examples of the reconstruction of the original hand posture using different numbers of principal components.

B. Hierarchical Optimization

While optimizing in a reduced PC-space yields plausible hand articulations, the estimation is limited to the postures contained in the used ground truth data set. To combine the benefits of reduced-DoF estimation with the flexibility of full-DoF estimation, we propose a hierarchical optimization scheme, in which a low-dimensional posture estimate is successively refined by incrementally adding DoFs during the estimation process.

This means that the IK optimization in the hand tracking process is performed in multiple stages, increasing the number of PC-parameters used in each stage. The early low-DoF estimation stages yield coarse initializations for the subsequent higher-DoF stages. In the final stage of this hierarchical scheme, all 26 DoFs are optimized. This allows for a coarse-to-fine optimization process that robustly refines the posture estimation based on hand synergies and yields highly accurate posture reconstructions. This approach also significantly lessens the dependency on temporal coherency in the input data. However, due to the increased computational complexity, this hierarchical scheme is thus far not suited for real-time tracking.

VI. Results

In this section we first illustrate the PCA of the captured hand posture data and give examples for the DoFs that can be represented in a reduced PC-space. Then, we present a detailed evaluation of the most important aspects of our system and show some results of our tracking approach.

A. PCA Results

We discuss the results of the PCA for two training data sets of hand postures: the complete data set of all recorded hand movements (including various manual interaction motions, sign language, and general hand movements) and a reduced data set containing only grasping movements. Approximately 80% of the variance in the complete data set is covered by the principal components associated with the largest 3 eigenvalues, and approximately 90% of the variance is covered by 6 principal components. For the grasping data set the 3 most significant principal components cover 90% of the variance. The 2 most significant principal components cover about 83% of the data variance and can already be used to represent meaningful hand synergies, which is illustrated in Figure 6. To clarify the information loss resulting from dimension reduction, Figure 7 shows the approximation error for a specific hand posture w.r.t. the number of dimensions used to represent it.

B. Evaluation with Synthetic Data

In order to evaluate the accuracy and overall performance of our hand tracking system we generated sequences of synthetic Kinect images using the virtual hand model. The model was animated using pre-recorded hand trajectories and synthetic depth images were rendered, which could then be used as input for our tracking system. This made it possible to compare the postures generated by our system with the known ground truth data.

The synthetic input data was generated with the same model that was also used for tracking; no noise was added to the depth images. This results in an idealized experimental setup, in which the potential for highly accurate posture reconstructions was given. We analyze the issues of noisy input data and mismatch between the tracked real hand and the virtual hand model in Section VI-C. Among the experiments we conducted are tests for evaluating the trade-off between the accuracy of the hand posture reconstruction and the runtime efficiency of the tracking, analyzing the benefits of using PC-space tracking, and the performance of single-frame posture estimation using hierarchical optimization. All experiments were conducted on an Octa-Core Intel Xeon(R) E5-1620 CPU at 3.60GHz with 8 GBs of RAM. Our implementation is heavily parallelized and fully utilizes all eight cores during the correspondence search and the construction of the Jacobian matrix.

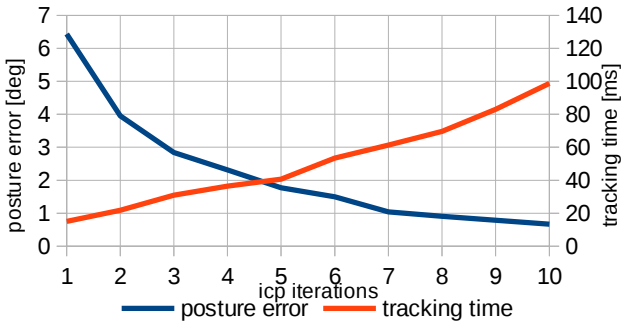


Fig. 8. Posture error and tracking time using subsampling density of 1/3 and varying numbers of non-rigid ICP iterations.

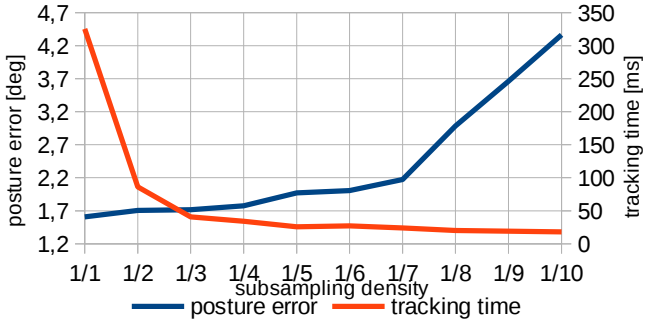


Fig. 9. Posture error and tracking time using five non-rigid ICP iterations and varying subsampling densities.

Accuracy-efficiency trade-off: The reconstruction accuracy increases with (i) the number k of point cloud-to-model correspondences used as input for the estimation and (ii) the number of non-rigid ICP iterations. However, in turn the runtime efficiency deteriorates with increased correspondences and more non-rigid ICP iterations.

In order to be able to track at interactive rates, we reduce the number of correspondences by linearly subsampling the input point cloud. The point cloud sampling density and the number of correspondence iterations are therefore the parameters that need to be adjusted to optimize the accuracy-efficiency trade-off. We tracked a synthetic input sequence with varying values for these parameters in order to find the best values for them. Evaluating the results for all combinations revealed that for our purposes a subsampling density of $1/3$ and five non-rigid ICP iterations provided the best trade-off between accuracy and efficiency. Figure 8 shows the posture error and tracking time for a subsampling density of $1/3$ and varying numbers of ICP iterations. After five iterations the error is lower than $2°$ and less than one third of the error of a single ICP iteration. The tracking time increases linearly in the number of iterations. Figure 9 shows the posture error and tracking time for five ICP iterations and varying subsampling values. The tracking time is approximately 40 ms at a subsampling density of 1/3, which facilitates sufficiently accurate tracking at approximately 25 fps.

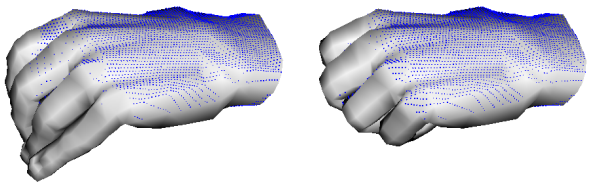
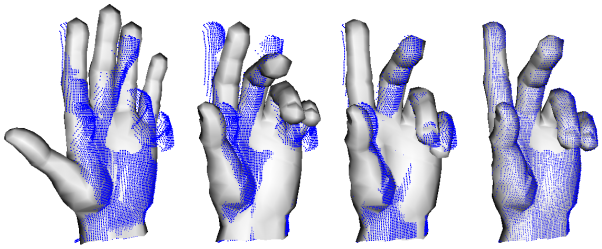


Fig. 10. Example for posture estimation with incomplete data. The blue points constitute the synthetically created input point cloud. The full-DoF estimation (left) cannot correctly reconstruct the grasp posture because the fingers in the input point cloud are partially occluded. The reduced-DoF estimation (right) is able to reconstruct the posture despite the missing data.

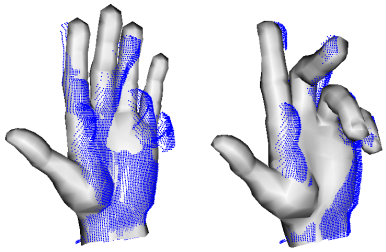
PCA-constrained tracking: The main benefits of PCA-constrained tracking are reduced computational complexity and improved posture reconstruction in cases of incomplete or ambiguous input data. To test this, we used a synthetic sequence of a grasping motion, in which the fingers are occluded by the back of the hand during the grasp and thus disappear from the input data. We tracked this sequence once with all DoFs and once with reduced DoFs based on PCA of the grasp training data set. The number of principal components (PCs) was chosen such that 99% of the variance in the data set was covered, which resulted in five PCs. In this experiment, the full-DoF tracking produced an average posture error of about $12°$, whereas the reduced-DoF tracking produced an average posture error of only about $4°$, despite the severe self occlusions present. The full-DoF estimation is less accurate because due to the partly missing correspondence data for individual fingers, the model's fingers stop moving or collapse into the wrong part of the input point cloud. The reduced-DoF estimation reconstructs the input data more accurately because the employed synergies cause the hand's DoFs to move in correlation to each other. This means that only some parts of the hand model need explicit correspondence data in order to approach an accurate reconstruction of the postures underlying the input. An example of this can be seen in Figure 10.

Thanks to the reduced number of DoFs, our PCA-based hand tracking improves runtime performance from 25 fps up to 30 fps in our experiments.

Hierarchical single-frame estimation: Since our hand tracking approach is dependent on temporal coherency in the input data, cases in which a good posture initialization is not known can be difficult to handle. The hierarchical optimization scheme described in Section V-B alleviates this dependency thanks to a coarse-to-fine estimation process. We confirmed this experimentally by performing single-frame posture estimation with and without hierarchical optimization. The input data for this experiment were several frames of distinct sign language gestures and the only initialization given was the global pose of the hand. The data set used for the PCA was the complete training set of recorded sign language hand trajectories. To facilitate maximally accurate reconstructions, we used the complete point cloud as input and performed 10 non-rigid ICP iterations. In cases of highly self-occluded input data it was impossible for the non-



(a) Hierarchical single-frame estimation



(b) Non-hierarchical single-frame estimation

Fig. 11. Example illustrating hierarchical and non-hierarchical single-frame posture estimation. The blue points constitute the synthetically created input point cloud. The hierarchical estimation successively approximates the input posture with increasing DoFs (from left to right: initial state, 1 DoF, 4 DoF, full DoF). The non-hierarchical estimation gets stuck in a local minimum due to bad initial correspondences (left: initial state, right: full DoF).

hierarchical full-DoF estimation to produce accurate posture reconstructions from a single frame, resulting in an average posture error of approximately $22°$ over all input frames. On the other hand, the hierarchical estimation arrived at an accurate reconstruction for *every* posture, resulting in an average posture error of approximately $0.4°$. The computation time for each frame was about 10s. Figure 11 shows a comparison between hierarchical and non-hierarchical single-frame estimation for a specific posture example.

C. Real-Time Hand Tracking

In practice, most of the idealized conditions of our experimental setup do not necessarily hold true, because the data obtained from the Kinect sensor can be noisy and the virtual hand model used for tracking usually does not perfectly match the user's real hand. We found that sensor noise did not severely impair the tracking quality within the distance of about 0.8 to 1.5 meters. To adjust to the hand sizes of different users we changed the overall scale of the hand model, which gave satisfactory results. This suggested that the geometric details of the virtual hand model are not that important during tracking, as long as the model's overall scale and proportions matched the user's hand.

We furthermore found that using a simplified cylinder-based hand model during tracking did not negatively affect the tracking accuracy and sped up the correspondence search. To evaluate the effect of using a cylinder hand model instead of a triangle mesh, we compared the posture reconstruction error using a cylinder model and the mesh model in our evaluation framework. Using full-DoF tracking the average posture error increased from approximately $2°$ to $4°$ using the cylinder model. Using reduced-DoF tracking there was virtually no difference in the posture error.

Figure 12 shows an example of a tracking sequence in which the full-DoF posture estimation results in an unnatural state due to a rapid hand motion causing incomplete sensor data, whereas the reduced-DoF estimation results in a natural approximation of the user's hand. This illustrates the benefit of performing the optimization in a reduced parameter space based on natural hand synergies.

Our tracking system runs at approximately 25 fps with full-DoF tracking and about 30 fps with $(6 + 6)$-DoF tracking in PC-space. A live performance of our real-time hand tracking system is shown in the accompanying video.

VII. Discussion

In our hand tracking approach, the hand posture is estimated using positional information from a Kinect point cloud as geometric constraints to fit a virtual hand model by means of inverse kinematics. The extension of the method to allow for optimization in a dimensionally reduced PC-space is simple and serves to constrain the parameter space in a meaningful way. Using a varied set of motion capture data as training data set facilitated the derivation of natural hand synergies that covered a wide range of natural hand movements.

Optimizing in a reduced PC-space as opposed to the high-dimensional hand posture space improves runtime performance and prevents implausible hand postures by constraining the estimation to realistic postures. These constraints can reduce the mobility of the posture estimation to some extent, but the overall tracking benefits from the increase in stability and plausibility of the estimated hand postures, especially in cases where the input data is incomplete or ambiguous.

The number of principal components needed to cover most of the variance in the data depends on the number and type of movements contained in the data set. The less varied the captured movements are, the less parameters are needed to represent the most meaningful hand synergies involved. This simplifies the problem of tracking specific movements, such as various types of grasping, to an optimization of only 2 or 3 posture parameters, but impairs the approximation of more general hand movements.

By employing a hierarchical optimization scheme, we are able to overcome the fact that low-DoF estimations are limited by the postures contained in the ground truth data. Our synergistic approach improves estimation quality by explicitly including prior knowledge about the tracked movements. If the types of movements are unknown prior to tracking or a very general posture estimation is desired, our purely IK-based approach yields satisfactory results unless the input data is highly inconsistent.

The main bottleneck of our system currently is the correspondence computation used in the non-rigid ICP optimization. We are planning on replacing our current CPU-based implementation with a GPU-based implementation. This way the complete input point cloud can be used during the fitting

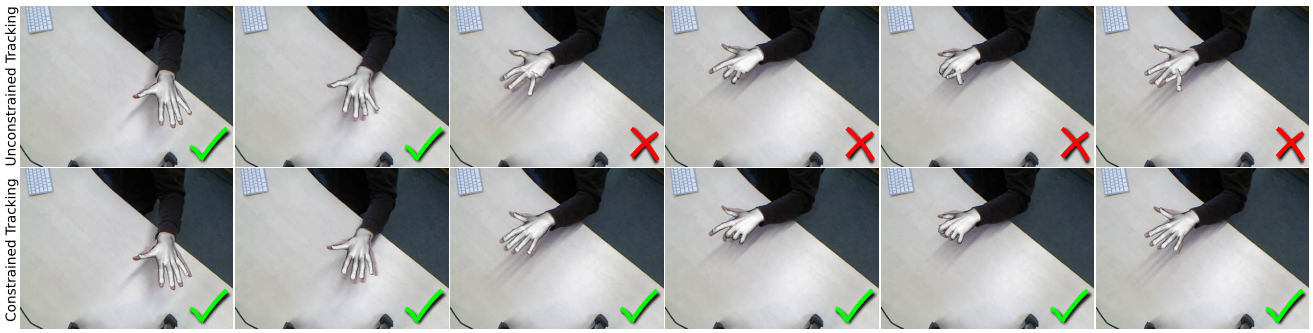


Fig. 12. Example hand tracking sequence. Top row: full-DoF posture estimation. Bottom row: reduced-DoF posture estimation. The full-DoF posture estimation fails to correctly track the posture during a rapid hand movement. The ring and pinky fingers collapse into the same point cloud segment, resulting in an unnatural posture. The reduced-DoF posture estimation has less mobility but maintains plausible hand postures across the whole sequence.

process without impairment of runtime performance, which will further improve estimation accuracy.

The accuracy of the approximation of the user's hand posture also increases when the dimensions of the virtual model closely match those of the user's hand. We are investigating a calibration procedure that automatically adjusts the virtual model's structural parameters, such as scale and finger segment lengths, to arbitrary hands.

ACKNOWLEDGMENT

This work was supported by the DFG Center of Excellence "Cognitive Interaction Technology" (CoE 277: CITEC) and the DFG grant "Real-Time Acquisition and Dynamic Modeling of Human Faces, Upper Bodies, and Hands" (BO 3562/1-1).

REFERENCES

[1] M. Schröder, J. Maycock, H. Ritter, and M. Botsch, "Analysis of hand synergies for inverse kinematics hand tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, Workshop: Hand synergies – how to tame the complexity of grasping.

[2] J. Romero, H. Kjellström, and D. Kragic, "Hands in action: real-time 3D reconstruction of hands in interaction with objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 458–463.

[3] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 63:1–63:8, 2009.

[4] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2088–2095.

[5] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys, "Motion capture of hands in action using discriminative salient points," in *European Conference on Computer Vision (ECCV)*, 2012, pp. 640–653.

[6] M. Schröder, C. Elbrechter, J. Maycock, R. Haschke, M. Botsch, and H. J. Ritter, "Real-time hand tracking with a color glove for the actuation of anthropomorphic robot hands," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2012, pp. 262–269.

[7] H. Hamer, K. Schindler, E. Koller-Meier, and L. V. Gool, "Tracking a hand manipulating an object," in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 1475–1482.

[8] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect," in *22nd British Machine Vision Conference (BMVC)*, 2011.

[9] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Tracking the articulated motion of two strongly interacting hands," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1862–1869.

[10] Y. Wang, J. Min, J. Zhang, Y. Liu, F. Xu, Q. Dai, and J. Chai, "Video-based hand manipulation capture through composite motion control," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 43:1–43:14, 2013.

[11] N. Bernstein, *The Co-ordination and Regulation of Movements*. Pergamon Press Ltd., 1967.

[12] M. Santello, M. Flanders, and J. F. Soechting, "Postural hand synergies for tool use," *Journal of Neuroscience*, vol. 18, no. 23, pp. 10 105–10 115, 1998.

[13] A. Daffertshofer, C. J. Lamoth, O. G. Meijer, and P. J. Beek, "PCA in studying coordination and variability: A tutorial," *Clinical Biomechanics*, vol. 19, no. 4, pp. 415–428, 2004.

[14] M. C. Tresch, V. C. Cheung, and A. d'Avella, "Matrix factorization algorithms for the identification of muscle synergies: evaluation on simulated and experimental data sets," *Journal of Neurophysiology*, vol. 95, no. 4, pp. 2199–2212, 2006.

[15] J. Lee and T. Kunii, "Model-based analysis of hand posture," *IEEE Computer Graphics and Applications*, vol. 15, no. 5, pp. 77–86, 1995.

[16] Y. Wu, J. Y. Lin, and T. S. Huang, "Capturing natural hand articulation," in *IEEE International Conference on Computer Vision (ICCV)*, 2001, pp. 426–432.

[17] M. Kato, Y.-W. Chen, and G. Xu, "Articulated hand motion tracking using ICA-based motion analysis and particle filtering," *Journal of Multimedia*, vol. 1, no. 3, pp. 52–60, 2006.

[18] M. Bray, E. Koller-Meier, and L. V. Gool, "Smart particle filtering for high-dimensional tracking," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 116–129, April 2007.

[19] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović, "Style-based inverse kinematics," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 522–531, 2004.

[20] A. Safonova, J. K. Hodgins, and N. S. Pollard, "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 514–521, 2004.

[21] D. Jacka, A. Reid, and B. Merry, "A comparison of linear skinning techniques for character animation," in *In Afrigraph*. ACM, 2007, pp. 177–186.

[22] S. R. Buss, "Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods," 2009, unpublished survey.

[23] S. R. Buss and J.-S. Kim, "Selectively damped least squares for inverse kinematics," *Journal of Graphics Tools*, vol. 10, no. 3, pp. 37–49, 2004.

[24] T. F. Chan and R. V. Dubey, "A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 2, pp. 286–292, 1995.

[25] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987.

[26] J. Maycock, D. Dornbusch, C. Elbrechter, R. Haschke, T. Schack, and H. Ritter, "Approaching manual intelligence," *Künstliche Intelligenz – Issue Cognition for Technical Systems*, vol. 24, no. 4, pp. 287–294, 2010.

[27] J. Maycock, J. Steffen, R. Haschke, and H. Ritter, "Robust tracking of human hand postures for robot teaching," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011, pp. 2947–2952.