Improved Estimation of Hand Postures Using Depth Images

Dennis Hamester, Doreen Jirak and Stefan Wermter

University of Hamburg, Department of Informatics, Knowledge Technology
Vogt-Kölln-Straße 30, D - 22527 Hamburg, Germany
{7hameste,jirak,wermter}@informatik.uni-hamburg.de
http://www.informatik.uni-hamburg.de/WTM/

***Abstract*—Hand pose estimation is the task of deriving a hand's articulation from sensory input, here depth images in particular. A novel approach states pose estimation as an optimization problem: a high-dimensional hypothesis space is constructed from a hand model, in which particle swarms search for the best pose hypothesis. We propose various additions to this approach. Our extended hand model includes anatomical constraints of hand motion by applying principal component analysis (PCA). This allows us to treat pose estimation as a problem with variable dimensionality. The most important benefit becomes visible once our PCA-enhanced model is combined with biased particle swarms. Several experiments show that accuracy and performance of pose estimation improve significantly.**

I. Introduction

The human hand is highly articulated. Humans use hands to manipulate objects in their surroundings and to communicate with other people. Capturing exact hand postures is an important step for Human-Robot Interaction and the development of natural interfaces. Computer vision (CV) can provide cheap and unobtrusive solutions to this problem, especially compared to data gloves.

Solving CV-based hand pose estimation without markers in single camera setups is a very challenging task, because hands can take on vastly different shapes in images. The amount of degrees of freedom (DOFs) contributes to a high-dimensional problem. The problem is further complicated by self-occlusions of the hand, that happen inevitably during the projection onto 2D images.

Following the taxonomy of Erol et al. [1], the approach discussed here belongs to the class of model-based tracking methods that follow a single hypothesis over time. In this context, single hypothesis means that only one satisfying solution is searched for and kept for the initialization of the next frame.

Significant progress in this area was made by Oikonomidis et al. [2]. They formulate pose estimation as an optimization problem. An internal hand model defines the parameters (DOFs) that make up a hand pose. This high-dimensional space is searched by a particle swarm for a suitable solution. As a particle moves, it renders an artificial depth image of its current hand pose hypothesis, which is compared to the actual observation from the Kinect. A target function is used to measure the discrepancy between rendered image and observation.

Oikonomidis et al.'s [2] method deals very well with the high-dimensionality and self-occlusions of the human hand. However, their approach is still computationally demanding. They report that their algorithm can run at about 15 FPS on a high-end PC. This is only half the rate at which the Kinect provides images. Our goal was to improve the performance, possibly to the point of running in real-time. At the same time we did not want to sacrifice any accuracy. We addressed this by exploiting biases in certain variants of particle swarms. We will show, that the optimization behavior of these variants can be aligned with a priori knowledge about how humans perform hand motions. The result was an overall improved convergence behavior, leading to better pose estimation in less time.

The idea to use a priori information has already been applied successfully to hand pose estimation by Bianchi et al. [3]. They determined statistical properties of hand motion and used these to improve the noisy measurements of a low-cost data glove. Our method differs in the way a priori knowledge is used. We use it to transform the search space of all hand postures, such that certain variants of particle swarm optimization (PSO) perform better due to biases in their behavior. We also do not require an existing pose estimation.

This paper is organized as follows: Section II covers our image preprocessing. Its purpose is to segment images into hand and non-hand parts. Section III introduces our new hand model and how its parameter space is altered through principal component analysis. Particle swarm optimization and the target function mentioned above are covered in Section IV. We will also explain our motivation for using a PSO variant with certain biases. In Section V we detail our experiments with the new method and provide an evaluation of the data. A final discussion and an outlook for future research are given in Section VI.

II. Hand Detection & Tracking

Detecting hands in images is a necessary step, because pose estimation is not capable of performing this segmentation by itself. We have separated the task into two steps: first, an initial one-time detection of hands based on depth images and shape recognition and second, subsequent tracking of the hand region with an adaptive skin color model.

For the first step, we restrict the detection to a specific hand posture that has a distinctive shape. A hand has to be open and face the sensor, with the fingers spread out a little. We perform foreground segmentation on depth images to reduce the region of interest. After that, edge detection in the foreground depth image provides a set of candidate contours. To support classification and the ability for generalization, we use Fourier descriptors with 12 complex-valued coefficients[1] to represent contours. These provide desirable invariance properties against common affine transformation (e. g. scale or rotation). Furthermore, the contour information is condensed in these 12 coefficients. Finally soft-margin support vector machines are used to separate hands from non-hands.

Based on the color that is enclosed by a hand contour, we learn the parameters of an elliptical boundary model (EBM) [4] of the skin color distribution. In all subsequent frames after successful detection by shape, this model is used to retrieve the hand.

This two-step scheme can properly distinguish between hands and other skin-colored objects in the scene. It comes at the cost of requiring a specific hand posture for detection. But this restriction is alleviated as soon as the distribution parameters of the skin color are learned.

III. Hand Model

The hand model serves two purposes: first, it defines the parameters that make up the state of the hand. Each parameter is one DOF of the model. Since our method requires synthetic hand images, the second purpose of our model is to define what a hand looks like. Apart from these, constraints of human hand articulations will be discussed as a third property.

A. Shape

The geometrical detail of our hand model must be kept low, while still ensuring resemblance to a real human hand. The algorithm repeatedly renders depth images, which are then compared to real depth images from the Kinect. Even though rendering is accelerated here with OpenGL, the complexity of the hand model has a huge impact on the runtime performance.

The model is shown in Fig. 1. It is composed of two primitive objects: elliptical cylinders and ellipsoids. The main object of the palm, shown in green in Fig. 1, is an elliptical cylinder, whose major semi-axis is significantly larger than the minor semi-axis. Two ellipsoids (blue) are placed on both ends of the cylinder to provide a smooth surface. Each finger consists of five objects: three cylinders and two spheres. The cylinders are shown in red, orange and yellow to emphasize the different phalanges. The spheres are placed between two adjacent phalanges to handle discontinuities that occur when bending a finger. The thumb is modelled similarly, but has a large ellipsoid (blue) as its first object instead of a cylinder. We found this to be very effective at reproducing the skin deformation that happens when the thumb is moved. Our model consists 27 individual objects: 15 elliptical cylinders and 12 ellipsoids.

B. Degrees of Freedom

A set of joints is placed into the above model, which allow the model to take on basically any articulation of a human hand. Joints come in two variants: joints with two DOFs and those with just one DOF.

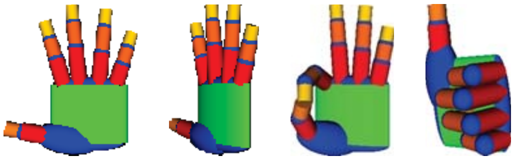


Fig. 1. Hand model in different configurations. It consists of two types of objects: ellipsoids (in blue) and elliptical cylinders (in green, red, orange and yellow).

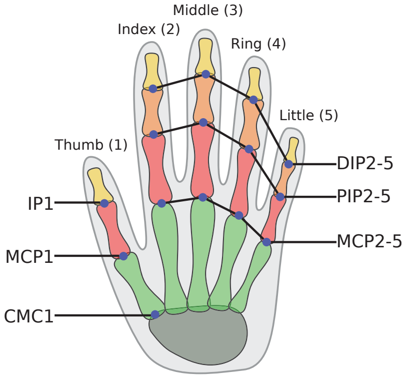


Fig. 2. Schematic joint model of a human hand. Joints are shown as blue dots. The CMC1 and MCP2-5 are modelled as 2-DOF joints. The other joints (MCP1, IP1, PIP2-5, DIP2-5) have only one DOF.

The joints used here describe a rotation, in either one or two dimensions. If it is a 2-DOF joint, both axes of rotation are orthogonal to each other and the reference point of the rotation is the same in both dimensions. This also implies that both axes must intersect, which is not necessarily true for real joints [5].

A schematic joint model of the human hand is depicted in Fig. 2. All DOFs together form a 20-dimensional parameter space, in which each point describes one particular posture. A special 6-DOF joint is placed in the center of mass of the palm, because we do not want restrict hands to one specific location in space nor is the orientation assumed fixed. This joint represent the global position and orientation of the hand in space relative to the sensor. With the addition of this joint, the final parameter space has 26 dimensions.

C. Constraints

One particular goal was to take inter-dependencies between DOFs into consideration. Although each joint has as many DOFs as stated before, we are not able to control all of them independently. Lin et al. [6] and Wu et al. [7] state, that 95% of the variance in hand articulations can be reduced to just 7 dimensions.

Lin et al. [6] further classify hand motion constraints into three types. The first type refers to static constraints, called *range of motion* values [5], for each individual DOF. They are usually expressed by two boundary angles that must not be exceeded. The second type refers to dynamic constraints that are caused by the anatomy of human hands. These can be

[1] The specific descriptor length was chosen on the basis of experiments.

TABLE I. VARIANCES AND (CUMULATIVE) RATIOS OF RANDOM HAND POSES AFTER PCA.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Variance	1.324	0.466	0.325	0.259	0.178	0.13	0.118	0.09	0.063	0.057	0.035	0.022	0.017	0.012	0.004	0.003
Ratio	42.7%	15%	10.5%	8.3%	5.7%	4.2%	3.8%	2.9%	2%	1.8%	1.1%	0.7%	0.5%	0.4%	0.1%	0.08%
Cum. Ratio	42.7%	57.7%	68.2%	76.5%	82.2%	86.5%	90.3%	93.2%	95.2%	97%	98.1%	98.9%	99.4%	99.8%	99.9%	100%

intra- and inter-finger constraints. One example for an intra-finger constraint is the fact that distal interphalangeal (DIP) and proximal interphalangeal (PIP) joints (refer to Fig. 2) can only be bent together [1], [6]. The third type comprises factors other than the anatomy, like e. g. the smoothness of hand motion.

We do not use closed formulas to model constraints, except for a few common type 2 constraints. These are the ones mentioned above, concerning the relationship between PIP and DIP angles. By using the equation

$$\theta_{DIP,i} = \frac{2}{3}\theta_{PIP,i} \quad 2 \leq i \leq 5 \tag{1}$$

the dimensionality is reduced by 4 down to 22.

We use PCA to further remove dimensions based on their significance. This also allows us to treat the dimensionality as a variable parameter instead of predefined value.

Several videos were recorded with the Kinect to generate the necessary data for the PCA. All videos together contained just above 9200 frames, which corresponded to about 5 minutes in total. Each video contained a sequence of mostly random finger motions, to capture as many hand postures as possible. Even though such motions are random in their articulation, they are still natural, in the sense that they are anatomically plausible, but lack semantic meaning. The hand was neither moved nor rotated in any of the videos. The palm always faced the sensor. In none of the videos, external physical forces were applied to the hand or fingers.

We used our hand model with 22 DOFs and estimated hand poses for each frame. The global position and orientation estimations were stripped from the resulting dataset and not considered for the PCA, since no meaningful correlation between global hand pose and finger articulation was expected. This left about 9200 estimations for the remaining 16 joint angles: the CMC1, MCP1-5, IP1 and PIP2-5 (refer to Fig. 2). The data had not been smoothed or filtered in any other way prior to the PCA.

The covariance matrix of the dataset is shown in Fig. 3. It shows some relatively high covariance values outside the diagonal. These give indication of the inter-dependencies between the DOFs. Interestingly, the DOFs that correspond to the abduction angles of the MCP joints (DOFs 4, 6, 8 and 10) do not show significant covariance values. The variances in Table I indicate very well, that much of the hand motion happens in only few dimensions. The data is similar to Lin et al. [6] and Wu et al. [7], in that 95% of the variance is concentrated in the first 9 dimensions (in both publications only 7 dimensions were required). The first two dimensions account for more than 50% of the variance. Based on this data, we assume that some of the least significant dimensions are essentially just noise.

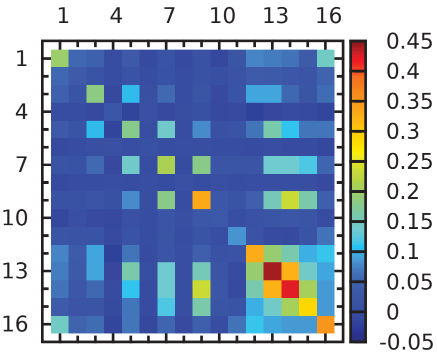


Fig. 3. Covariance matrix of random hand poses. The DOFs are: 1,2=CMC1; 3,4=MCP2; 5,6=MCP3; 7,8=MCP4; 9,10=MCP5; 11=MCP1; 12=PIP2; 13=PIP3; 14=PIP4; 15=PIP5; 16=IP. Position, orientation and DIP joints are not included.

IV. PARTICLE SWARM OPTIMIZATION

Particle swarms were developed in 1995 by Kennedy and Eberhart [8] and have received great attention since then [9], [10]. The method originates from human social behavior simulations. In these simulations, agents were placed in a two-dimensional space and moved through it in discrete time steps. The direction of the movement was based on an attraction point, which Kennedy and Eberhart [8] called the *cornfield vector*, in analogy to bird flocks searching for food. The authors observed that all agents settled quickly on the attraction point, despite their random initialization. The result was the formulation of the original particle swarm optimization algorithm.

In a *swarm*, n simple entities, called *particles*, exist in a d-dimensional space. Each particle has a position $p \in \mathbb{R}^d$ and velocity $v \in \mathbb{R}^d$. When particles move in discrete steps over time, they evaluate their own position p with a target function f (which will be detailed shortly). The goal is to minimize this function. After all particles moved, their velocities are updated. The new velocity comprises two distinct components: a cognitive and a social component [11]. The cognitive component is the position $p_{c,i}$, with the best target value a particle i has seen in the past. As such, the cognitive component potentially differs between all particles. The social component on the other hand, is the global best known position p_s and is shared between all particles in the swarm. These two vectors take on the role of attraction points in the following formula for the velocity update:

$$\begin{aligned} v_i \leftarrow \chi\,[\; & v_i + U(0,\phi_c) \otimes (p_{c,i} - p_i) \\ & + U(0,\phi_s) \otimes (p_s - p_i)\;] \end{aligned} \tag{2}$$

$U(a,b)$ is a vector of d random numbers, each uniformly distributed in the range given by its parameters and $\otimes$ denotes component-wise multiplication. The parameters ϕ_c and ϕ_s control the influence of the cognitive and social component on the new velocity. The parameter χ is used to control the velocities and avoid swarm-explosion. It can be computed as follows [12]:

$$\begin{aligned}\phi &= \phi_c + \phi_s > 4\\ \chi &= \frac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}}\end{aligned} \tag{3}$$

For each particle i, the positions are then updated by simply adding the velocity:

$$p_i \leftarrow p_i + v_i \tag{4}$$

The target function we use here determines how closely a given hand pose hypothesis $h \in \mathbb{R}^{26}$ matches the observation depth image d_o. Let $x \times y$ be the image size and d_h the rendered depth image of h using our hand model. Then the function

$$f(h) = \sum_{v=1}^{y} \sum_{u=1}^{x} \min\left(\left|d_o(u,v) - d_h(u,v)\right|, t\right) \tag{5}$$

iterates over both images and computes the sum of pixel-wise differences, which are thresholded at some value t. Areas in both images that do not contain hand pixels, are marked with a value of 0. If we omitted the threshold, this would lead to very high numbers caused by possibly few pixels. We experimentally determined $t = 5\text{cm}$ to work well.

This function was designed similarly to the one proposed by Oikonomidis et al. [2], but also radically simplified. Originally, two more components were included. First, a term that tested whether a pixel is skin-colored or not. This would be redundant in our case, because all other pixels in d_o have already been filtered out during the tracking phase (Section II). The second term penalized physically implausible hand postures. More specifically, it considered the differences in abduction angles of the three adjacent finger pairs. In our experiments, we observed that such a term actually hinders proper optimization. Most of the cases in which this happened, had relative abductions close to zero between adjacent fingers (like in a stop gesture or a fist). We therefore removed this penalizing term.

Spears et al. [13] showed, that drawing random numbers dimension by dimension in equation (2) causes several biases. They found out, that the bias is made up of two components: *skew* and *spread*. When a particle moves primarily parallel to an axis, the skew bias pushes it towards a diagonal of two or more axes. On the other hand, a particle that moves along a diagonal is highly unstable and gets pushed back to a trajectory parallel to an axis due to the spread bias. The biases appeared regardless of PSO parameters, like swarm size, number of iterations and dimensionality. A particle swarm that updates velocities dimension by dimension is biased towards movement along axis parallels, even when the problem is rotationally symmetric. As a direct result, new PSO versions (like SPSO 2011 [11]) were developed to overcome the bias.

We deliberately propose to use PSO *with* these biases. In our application, particles move through the parameter space of our hand model, in which each axis corresponds to one DOF. However as detailed in Section III, this space is altered by a PCA. The PCA rotates the hand model's parameter space in such a way that eigenvectors become the new coordinate axes. These new axes do no longer represent just one DOF, but many. Specifically, each axis models a particular (linear) finger motion involving possibly many DOFs that was (with decreasing significance) noticeable in the sample data. Consider for example the motion between an open hand and a fist. In the original space, this requires changing many DOFs at the same time. If we further assume that this motion corresponds roughly to one of the principal components, this might just involve change in a very limited subset of DOFs, after the space has been rotated by the PCA. In this regard, the new parameter space is aligned with the way particles move in a biased PSO. Most significant changes in hand postures happen along parallels to coordinate axes, and less likely along a mix of many axes (diagonals).

The switch to a biased PSO in combination with PCA revealed another positive side effect during our experiments, besides the increase in accuracy. Oikonomidis et al. [2] were forced to randomly disturb the particles every few generations due to premature convergence ("swarm collapse"). At first, we observed the same behavior. However, after making the discussed changes, the swarm collapses disappeared. With our method, the swarm does not converge prematurely and is able to find satisfying solutions on its own.

V. Experiments & Evaluations

Our goal for the experiments and evaluations was to assess the differences of our pose estimation compared to Oikonomidis et al. [2]. We were primarily interested in measuring the possible accuracy gains through quantitative evaluation. We will also present some qualitative results at the end of this section.

Evaluating a hand pose estimation method is in itself not a trivial task, because ground-truth information is not available when working with real videos. To deal with this problem, we generated a test video, in part synthetically. The images, that make up the video, were completely rendered with our hand model, but the actual movement of the hand was authentic. We first recorded a real video of the desired hand motions and then ran the hand pose estimation on them. Gaussian filters were used to eliminate high frequency noise in the hand pose sequence. This filtered sequence was then used in turn to render the synthetic video of the hand motions. As a last step, we applied noise and a discretization step to the video in order to mimic depth images from the Kinect.

The video contained random motions of the fingers and the thumb and lasted for approximately 25 seconds. We tried to capture movement of all DOFs and cover many possible articulations. This video did not contain notable movement of the whole hand in space. The hand itself was also not rotated. For the whole duration of the video, the palm was facing the sensor. The mean distance between sensor and hand was roughly one meter.

Let $\pi_i(x)$ be the projection of the vector x onto its i-th component and $x_1, x_2 \in \mathbb{R}^{26}$ be hand poses. Then

$$e_a(x_1, x_2) = \frac{1}{23} \sum_{i=4}^{26} \left|\pi_i(x_1 - x_2)\right| \tag{6}$$

measures the discrepancy of all angles (given in degrees $[^\circ]$) as the mean absolute difference. The first three components

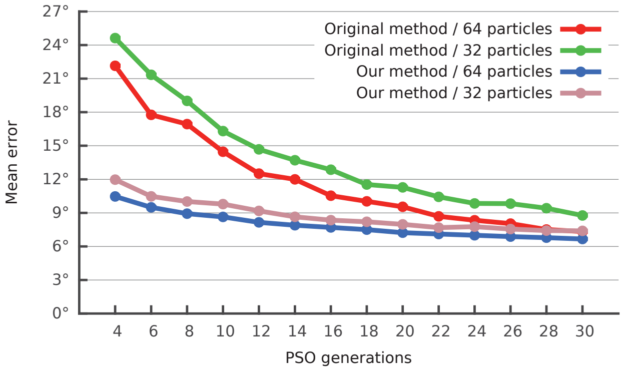


Fig. 4. Comparison of our method (blue, magenta) with Oikonomidis et al. [2] (red, green). A single measurement indicates the angle error (equation 6) averaged over all frames in the test video.

correspond to the hand location in 3D space, which was not considered for evaluation. In contrast to Oikonomidis et al. [2], we chose to stay close to the actual representation of hand poses as a vector of mostly angles. They derived locations of phalanx endpoints and used them to measure accuracy.

This evaluation did not take into consideration PSO parameters other than the number of particles and generations. In particular, the effect of the cognitive and social factors in the particles velocity equation (2) was not analyzed. To conduct the experiments, we set the values to $\phi_c = 2.8$ and $\phi_s = 1.3$ [2], i.e. the constriction factor χ was 0.73 (equation 3). Most combinations for ϕ_c and ϕ_s perform well, as long as $\phi_c + \phi_s = 4.1$ holds true [14].

A. Direct Comparison

The vertical axis in Fig. 4 shows the mean absolute angle error e_a. A single measurement is the mean error over all frames in the entire video for the given PSO parameters. The original method shows a strong dependency on the number of generations. To keep the error below an average of 9° at least 64 particles and 22 generations had to be used. For our method on the other hand, 32 particles and 14 generations already were sufficient. In general, we observed much faster convergence after enabling the PCA and biased PSO. The curves for our method are less steep in Fig 4. This directly translates to an improved performance, because less effort was required to achieve a certain maximum error. Using 64 particles and 25 generations has been suggested before [2]. We reached the same error at 32/18, which is roughly 2.8 times faster.

B. Dimensionality Reduction

The experiments above used our hand model with 22 DOFs. We applied the PCA but did not remove any dimensions afterwards. Figure 5 depicts the same experiment (32 particles, 20 generations) but with a varying number of dimensions. The lowest possible number of dimensions is 7, which include 6 for the global pose and just one dimension for all joint angles. The data indicate that there was no benefit in removing dimensions. Starting from the right, the mean error first stagnates and then starts rising. Thus, our method performs optimally when all dimensions are left in place. The biased PSO did not seem to be influenced negatively when insignificant dimensions were present.

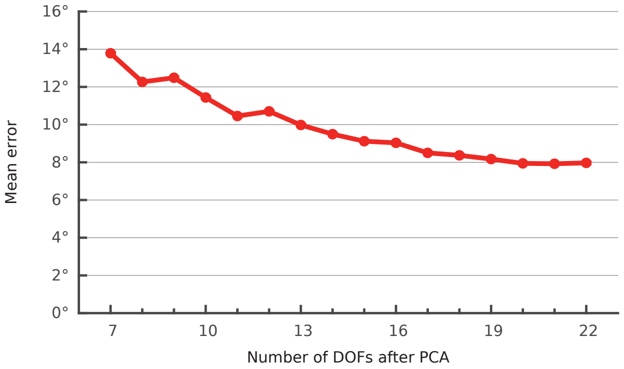


Fig. 5. Dependency between DOFs and mean error.

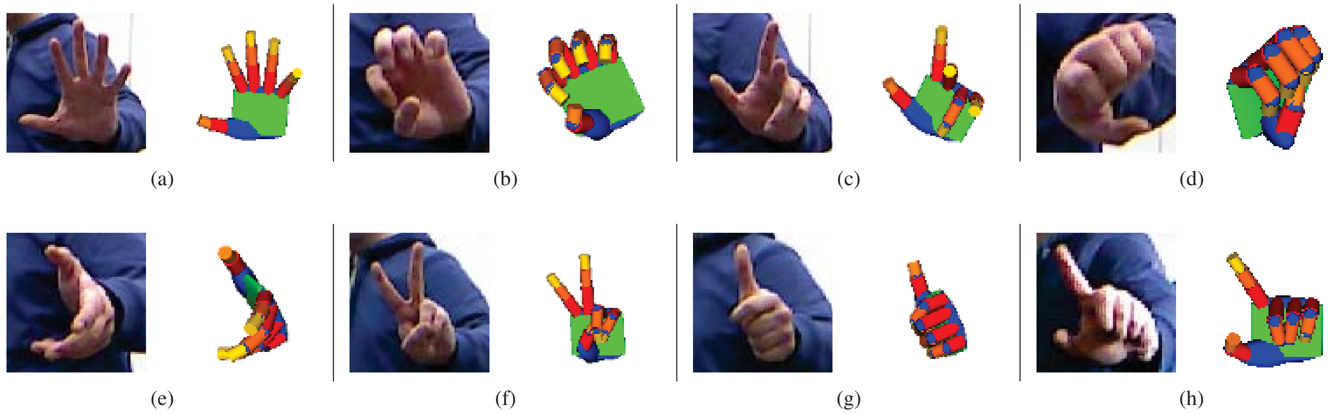
C. Qualitative Results

When it comes to the visually perceived accuracy of our pose estimation, there were only minor discrepancies compared to the real hand posture. Figure 6 shows eight different postures alongside the model, articulated according to the estimation. Most errors stem from thumb estimation. Particularly in Fig. 6(d), the thumb does not point away from the hand. It is actually inside the other fingers, which is also the case in Fig. 6(f). This happened quite often, because we did not perform collision detection or model any physical constraints. Figures 6(e) and (g) show postures with severe out-of-plane rotations, that still resulted in proper estimations. We have found these kinds of postures to be especially problematic, because the hand occludes large parts of itself.

VI. Discussion & Future Work

In this paper we presented an improved method for the problem of full-DOF hand pose estimation, based on the method by Oikonomidis et al. [2] that has been extended to take a priori knowledge about hand motion into consideration. We achieved this by first applying a common relationship between DIP and PIP joints, followed by a change of basis to eigenvectors. This way, biases in particle swarms can be exploited, leading to much improved convergence behavior. We performed several experiments with partially synthetic data to provide evidence for this claim. Other experiments revealed that our method retains its optimal accuracy when all dimensions are

We discussed our hand model from three different perspectives: shape, DOFs and constraints. Several similar works [2], [15], [16] focus almost exclusively on the shape of the model, while we put more emphasis on the DOFs of the model instead. With the terminology of Lin et al. [6], only level 1 constraints have been imposed on joint angles in the relevant literature [2], [15], [16]. In this paper, constraints of level 2 and 3 were considered, and some have been modelled with closed formulas, while the majority is included through PCA. This also introduced the dimensionality of the hand model as a parameter instead of a fixed value. The PCA played a major role in the improved properties of our method.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 6. Eight hand postures and their estimation.

Biased particle swarms were the other key component in gaining accuracy. We recognized that these biases push particles onto trajectories parallel to coordinate axes and how this relates to PCA. The particle swarm in our method does not converge prematurely. We were thus not forced to apply additional randomness to keep the swarm alive, like Oikonomidis et al. [2] did.

A. Contributions

Our method maintains the same level of accuracy as before [2], but is about 2.8 times faster. If a 16% increase in estimation errors is acceptable, our method is able to run five times faster. It is able to exceed the 30 Hz framerate of the Kinect. We expect that even more performance is achievable with our method when the set of possible hand postures is constrained by specific applications. Our idea to combine biased PSO with PCA provides a very flexible way of incorporating a priori knowledge.

We also gave a working example on how biased PSO algorithms can be exploited and that the results can be significant. This might prove useful to many more applications of PSO, because it is not specific to pose estimation.

B. Future Work

Despite our improvements, the approach is still computationally demanding. Most of the time is spent rendering depth images. For future work, we would like to explore ways of shifting some of the effort to an offline learning phase. This might be done by pre-rendering a subset of hand postures and a suitable interpolation method.

We plan to also conduct additional experiments to identify circumstances under which the algorithm fails. First experiments indicate that out-of-plane rotations of the palm require significantly more effort for a proper pose estimation. These rotations are characterized by the palm not being aligned with sensor image plane, as in Figs. 6(e) and (g).

Acknowledgment

The authors would like to thank Johannes Bauer and Sven Magg for providing valuable suggestions when writing this paper. We would also like to thank the anonymous reviewers for their valuable comments on our paper.

References

[1] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, pp. 52–73, 2007.

[2] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3D tracking of hand articulations using kinect," in *British Machine Vision Conference, Proc. of the*, 2011, pp. 101.1–101.11.

[3] M. Bianchi, P. Salaris, and A. Bicchi, "Synergy-based hand pose sensing: Reconstruction enhancement," *Int. Journal of Robotics Research, The*, vol. 32, no. 4, pp. 396–406, 2013.

[4] J. Y. Lee and S. I. Yoo, "An elliptical boundary model for skin color detection," in *Imaging Science, Systems, and Technology, Int. Conf. on*, 2002.

[5] G. Stillfried and P. van der Smagt, "Movement model of a human hand based on magnetic resonance imaging (mri)," in *Applied Bionics and Biomechanics, Int. Conf. on*, 2010.

[6] J. Lin, Y. Wu, and T. S. Huang, "Modeling the constraints of human hand motion," in *Human Motion, Proc. Workshop on*, 2000, pp. 121–126.

[7] Y. Wu, J. Y. Lin, and T. S. Huang, "Capturing natural hand articulation," in *Computer Vision, IEEE Int. Conf. on*, vol. 2, 2001, pp. 426–432.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, IEEE Int. Conf. on*, vol. 4, 1995, pp. 1942–1948.

[9] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization - an overview," *Swarm Intelligence*, vol. 1, pp. 33–57, 2007.

[10] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," *Journal of Artificial Evolution and Applications*, vol. 2008, pp. 3:1–3:10, 2008.

[11] S. S. Pace, A. Cain, and C. J. Woodward, "A consolidated model of particle swarm optimisation variants," in *Evolutionary Computation, IEEE Congress on*, 2012, pp. 1–8.

[12] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, 2002.

[13] W. M. Spears, D. Green, and D. F. Spears, "Biases in particle swarm optimization," *Int. Journal of Swarm Intelligence Research*, vol. 1, no. 2, pp. 34–57, 2010.

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

[15] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool, "Tracking a hand manipulating an object," in *Computer Vision, IEEE Int. Conf. on*, 2009, pp. 1475–1482.

[16] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Consumer Depth Cameras for Computer Vision*, ser. Advances in Computer Vision and Pattern Recognition. Springer London, 2013, pp. 119–137.