Hand Tracking with an Extended Self-Organizing Map

Andreea State[1,2], Foti Coleca[1,3], Erhardt Barth[1,3], and Thomas Martinetz[1]

[1]Institute for Neuro- and Bioinformatics, University of Lübeck
Ratzeburger Allee 160, 23538 Lübeck, Germany
{state,coleca,barth,martinetz}@inb.uni-luebeck.de
http://www.inb.uni-luebeck.de
[2]University "POLITEHNICA" of Bucureşti
Splaiul Independenţei 313, 060042 Bucureşti, Romania
http://www.upb.ro
[3]Gestigon GmbH, Innovations Campus Lübeck
Maria-Goeppert Straße 1, 23562 Lübeck, Germany
http://www.gestigon.de

Abstract. We introduce an extension of the self-organizing map for performing 3D hand skeleton tracking. We use a range camera for data acquisition and apply a SOM-like learning process within each frame in order to capture the hand pose. Our method uses a topology consisting of 1D and 2D segments for an improved representation of the hand. The proposed algorithm is very efficient and produces good tracking results.

Keywords: hand skeleton tracking, self-organizing maps, kinect.

1 Introduction

The problem of object tracking and pose estimation has gained much attention during the last years, due to the variety of new technologies and devices designed for 3D image acquisition. While standard 2D cameras necessitate more complex image processing techniques, 3D cameras provide a more favorable framework for tracking algorithms, as depth information enables the reconstruction of 3D objects. In particular, hand tracking can be used in a wide variety of applications, e.g. gesture recognition, and represents a milestone in human-machine interaction. The difficulty lies in the fact that the state space is extremely large, due to the 27 degrees of freedom of the human hand [1].

Published in: P. A. Estvez, J. C. Prncipe, P. Zegers (Eds.), Advances in Self-Organizing Maps, vol. 198 of Advances in Intelligent Systems and Computing, Springer Berlin Heidelberg, ISBN 978-3-642-35229-4, 115–124, doi:10.1007/978-3-642-35230-0 12, URL http://dx.doi.org/10.1007/978-3-642-35230-0_12, 2013.

Our work focuses on developing a hand tracking algorithm for 3D cameras. Having acquired 3D image information, we aim at building a tracking algorithm that is both accurate and of low computational cost. This is achieved with an extension of the approach introduced in [10]. After a simple preprocessing step which assumes that the hand is the closest object to the camera, we obtain a point cloud in 3D of the hand which we then represent with Kohonen's self-organizing map [3]. For this purpose the topology of the Kohonen network is crafted according to the skeleton of a hand, as illustrated in Fig. 1. However, to be able to obtain good tracking results, we have to extend the SOM algorithm such that the point cloud of the hand is represented not only by the nodes of the network, but also by the line and plane segments between the nodes.

Compared to our approach based on a SOM, the authors of [4] use kinematic models and build a hand state model which consists in a set of lines and points generated by the projection of the hand model into the image plane. Hand pose estimation based on features derived from projections of the hand and its shadow is presented in [5]. Nevertheless, the method requires controlled background and lighting, and is susceptible to occlusion. The authors of [6] introduce a machine learning architecture for matching image features to 3D hand example poses, through solving an optimization problem based on Bayes' rule. Another approach is to estimate hand pose with a database of synthetic hand images. For instance, in [7] an indexed image database is used to retrieve the closest hand match, with an adapted chamfer distance and line matching algorithm. In [8], the authors implement a cascade of increasingly complex classifiers to determine hand pose from synthetic training data. In order to handle occlusion, particle filters can be used. In [9], the authors apply a meta-descent algorithm to minimize the distance between a predicted position and the observed position, while particle filters predict new sample positions and help the optimization algorithm to recover from local minima. As shown in [2], the combined usage of intensity images and range information provides a good framework for body tracking.

Section 2 will consist in a detailed explanation of our method, followed by evaluation and results in Section 3. Finally, we present a few conclusions in Section 4.

2 The Extended SOM

We use a network topology that can suitably describe a human hand. Our network is made up of sixteen nodes, and the defined connections are illustrated in Fig. 1.

The standard SOM algorithm starts with the initialization of the network weights, followed by the iteration of two procedures, the competition and the update of the weights. At each iteration, a sample point from the dataset is randomly chosen. During the competition phase, the algorithm determines a winner node, that is, the node characterized by the weight with the minimum Euclidean distance from the sample point.

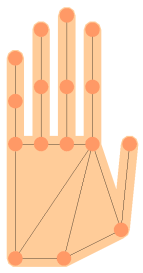

Fig. 1. The extended SOM left-hand topology

Given a network with n neurons and a sample point $\mathbf{x}$, we determine the winner node $\hat{i}$ as follows:

$$\hat{i} = arg\{\min_i ||\mathbf{x} - \mathbf{w}_i||_2\},\ i = 1, \dots, n \tag{1}$$

with $\mathbf{w}_i$ being the weight of node i. Next, the update phase aims at decreasing the distance between the winner-node weight and the sample point, by an amount given by the learning rate $\epsilon(t)$. First, let us define the learning rate function as

$$\epsilon(t) = \epsilon_i \left(\frac{\epsilon_f}{\epsilon_i}\right)^{\frac{t}{t_{max}}}, \tag{2}$$

where ϵ_i is the initial learning rate, ϵ_f is the final learning rate, t is the current iteration and t_{max} is the maximum number of iterations performed on the network. Then, the weight $\mathbf{w}_{\hat{i}}$ is updated at step t according to:

$$\mathbf{w}_{\hat{i}}(t+1) = \mathbf{w}_{\hat{i}}(t) + \epsilon(t)(\mathbf{x} - \mathbf{w}_{\hat{i}}(t)) . \tag{3}$$

The standard SOM algorithm also uses a neighbourhood update, in the sense that not only the winner-node weight is updated, but also the weights of the neighbour-nodes.

Our proposed algorithm extends the competition and the update step to 1D and 2D network segments. The 1D-segments are the lines between pairs of connected nodes, and the 2D-segments are the triangles determined by triples of connected nodes. 1D-segments allow to represent the fingers more accurately, and the 2D-segments form the palm of the hand. By segment updates we aim at minimizing the average distance between network segments and points from the dataset (point cloud of the hand provided by the 3D-camera). Now we not only have elements of dimension zero (nodes) like in the classical case, but also elements of dimension one and two for representing the data distribution. This approach is motivated by the fact that a hand-like topology involves a difficult separation between the nodes corresponding to fingers. A node that belongs to one finger can easily be attracted by another finger, given the topological closeness. This may lead to an erroneous tracking of the hand and destroy the topological relations. With these 1D and 2D segments we can represent fingers and parts of the palm more accurately and expect the self-organizing maps to be less prone to this type of errors.

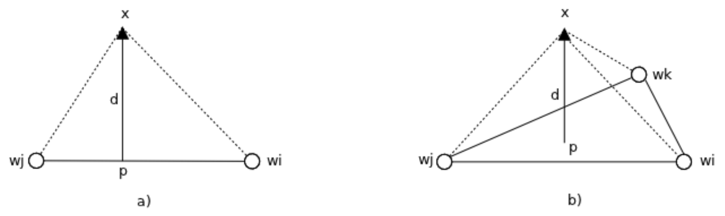

Fig. 2. a) The projection $\mathbf{p}$ of sample point $\mathbf{x}$ onto 1D-segment $[\mathbf{w}_i; \mathbf{w}_j]$. b) The projection $\mathbf{p}$ of sample point $\mathbf{x}$ onto 2D-segment $[\mathbf{w}_i; \mathbf{w}_j; \mathbf{w}_k]$.

The competition phase in our extended SOM algorithm determines whether a single node or a 1D-segment or a 2D-segment is closest to the randomly chosen sample point $\mathbf{x}$. Depending on this result, the update phase will either perfom a classical node update as shown in Equation 3 or a segment update.

The distance to a 1D-segment $[\mathbf{w}_i; \mathbf{w}_j]$ (see Fig. 2a) is determined via the projection point $\mathbf{p}$ of $\mathbf{x}$ onto the given segment. Let us define $\mathbf{d}_{ji} = \mathbf{w}_j - \mathbf{w}_i$ and $\mathbf{d}_{ij} = \mathbf{w}_i - \mathbf{w}_j$. Then, we may write $\mathbf{p}$ as

$$\mathbf{p} = \mathbf{w}_i + \eta_{ji}\mathbf{d}_{ji},\ 0 \leq \eta_{ji} \leq 1\,. \tag{4}$$

Similarly,

$$\mathbf{p} = \mathbf{w}_j + \eta_{ij}\mathbf{d}_{ij},\ 0 \leq \eta_{ij} \leq 1 \tag{5}$$

and $\eta_{ji} + \eta_{ij} = 1$. Given the unit vectors $\hat{\mathbf{d}}_{ji}$ and $\hat{\mathbf{d}}_{ij}$, the coefficients η_{ji} and η_{ij} are

$$\eta_{ji} = \frac{(\mathbf{x} - \mathbf{w}_i)\hat{\mathbf{d}}_{ji}}{||\mathbf{d}_{ji}||} \tag{6}$$

$$\eta_{ij} = \frac{(\mathbf{x} - \mathbf{w}_j)\hat{\mathbf{d}}_{ij}}{||\mathbf{d}_{ij}||}\,. \tag{7}$$

Then the squared distance $||\mathbf{d}||^2$ of $\mathbf{x}$ to the 1D-segment $[\mathbf{w}_i; \mathbf{w}_j]$ is

$$\begin{aligned} ||\mathbf{d}||^2 &= ||\mathbf{x} - \mathbf{p}||^2 \\ &= ||\mathbf{x} - \mathbf{w}_i||^2 - ||\mathbf{p} - \mathbf{w}_i||^2 \\ &= ||\mathbf{x} - \mathbf{w}_i||^2 - \eta_{ji}^2||\mathbf{w}_j - \mathbf{w}_i||^2\,. \end{aligned} \tag{8}$$

The 1D-segment that is closest to $\mathbf{x}$ is determined by

$$(\hat{\imath}, \hat{\jmath}) = arg\{\underset{ij}{min}||\mathbf{d}_{ij}||\}, i, j = 1, \ldots, n\,. \tag{9}$$

Evidently, the above equation applies only to pairs of connected nodes (i, j).

Similarly, the distance to a 2D-segment (see Fig. 2b) can be determined. Let us define $\mathbf{d}_{ji} = \mathbf{w}_j - \mathbf{w}_i$, $\mathbf{d}_{ki} = \mathbf{w}_k - \mathbf{w}_i$. Then, we may write $\mathbf{p}$ as

$$\mathbf{p} = \mathbf{w}_i + \eta_{ji}\mathbf{d}_{ji} + \eta_{ki}\mathbf{d}_{ki},\ 0 \le \eta_{ji}, \eta_{ki} \le 1,\ \eta_{ji} + \eta_{ki} \le 1\,. \tag{10}$$

Analogously,

$$\mathbf{p} = \mathbf{w}_j + \eta_{ij}\mathbf{d}_{ij} + \eta_{kj}\mathbf{d}_{kj},\ 0 \le \eta_{ij}, \eta_{kj} \le 1,\ \eta_{ij} + \eta_{kj} \le 1 \tag{11}$$

and

$$\mathbf{p} = \mathbf{w}_k + \eta_{ik}\mathbf{d}_{ik} + \eta_{jk}\mathbf{d}_{jk},\ 0 \le \eta_{ik}, \eta_{jk} \le 1,\ \eta_{ik} + \eta_{jk} \le 1\,. \tag{12}$$

We then compute the squared distance $||\mathbf{d}||^2$ of $\mathbf{x}$ to the 2D-segment (triangle) determined by $[\mathbf{w}_i; \mathbf{w}_j; \mathbf{w}_k]$ according to

$$\begin{aligned} ||\mathbf{d}||^2 &= ||\mathbf{x} - \mathbf{p}||^2 \\ &= ||\mathbf{x} - \mathbf{w}_i - \eta_{ji}\mathbf{d}_{ji} - \eta_{ki}\mathbf{d}_{ki}||^2 \end{aligned} \tag{13}$$

where

$$\eta_{ji} = \frac{(\mathbf{x} - \mathbf{w}_i)\hat{\mathbf{d}}_{ji} - \Big((\mathbf{x} - \mathbf{w}_i)\hat{\mathbf{d}}_{ki}\Big)\,(\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})}{||\mathbf{d}_{ji}||\,\Big(1 - (\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})^2\Big)} \tag{14}$$

$$\eta_{ki} = \frac{(\mathbf{x} - \mathbf{w}_i)\hat{\mathbf{d}}_{ki} - \Big((\mathbf{x} - \mathbf{w}_i)\hat{\mathbf{d}}_{ji}\Big)\,(\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})}{||\mathbf{d}_{ki}||\,\Big(1 - (\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})^2\Big)}\,. \tag{15}$$

After having determined whether one of the nodes, a 1D-segment or a 2D-segment is closest to the randomly chosen sample point, the update procedure takes place. The simplest situation is illustrated in Fig. 3a, when a single node is closest and has to be updated. This is done according to the standard SOM algorithm. In case a segment has to be updated, the nodes which determine the segment have to be moved such that the distance $||\mathbf{d}||$ is reduced. We derive the respective movement by gradient descent minimization on the squared segment distance. For node j we obtain

$$\begin{aligned} -\frac{1}{2}\frac{\partial \mathbf{d}^2}{\partial \mathbf{w}_j} &= -\frac{1}{2}\frac{\partial}{\partial \mathbf{w}_j}\left((\mathbf{x} - \mathbf{w}_i)^2 - \eta_{ji}^2(\mathbf{w}_j - \mathbf{w}_i)^2\right) \\ &= \frac{1}{2}\frac{\partial}{\partial \mathbf{w}_j}\left(\eta_{ji}^2(\mathbf{w}_j - \mathbf{w}_i)^2\right) \\ &= \frac{1}{2}\left(\frac{\partial}{\partial \mathbf{w}_j}\eta_{ji}^2\right)(\mathbf{w}_j - \mathbf{w}_i)^2 + \frac{1}{2}\eta_{ji}^2\left(\frac{\partial}{\partial \mathbf{w}_j}(\mathbf{w}_j - \mathbf{w}_i)^2\right) \\ &= \eta_{ji}\left(\frac{\partial}{\partial \mathbf{w}_j}\eta_{ji}\right)(\mathbf{w}_j - \mathbf{w}_i)^2 + \eta_{ji}^2(\mathbf{w}_j - \mathbf{w}_i) \\ &= \eta_{ji}\left[(\mathbf{x} - \mathbf{w}_i) - 2\frac{(\mathbf{x} - \mathbf{w}_i)(\mathbf{w}_j - \mathbf{w}_i)}{(\mathbf{w}_j - \mathbf{w}_i)^2}(\mathbf{w}_j - \mathbf{w}_i)\right] + \eta_{ji}^2(\mathbf{w}_j - \mathbf{w}_i) \end{aligned}$$

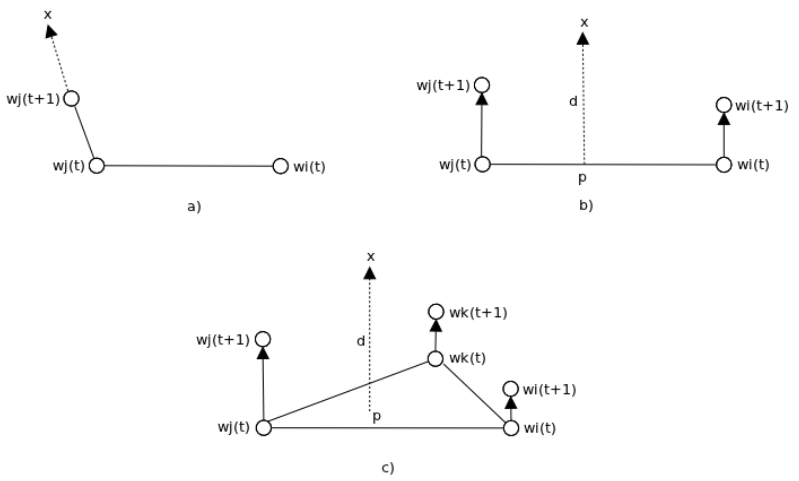

Fig. 3. a) Weight $\mathbf{w}_j$ is displaced towards the data point $\mathbf{x}$, on the direction given by vector $\mathbf{x}-\mathbf{w}_j$. b) Weights $\mathbf{w}_i$ and $\mathbf{w}_j$ are displaced towards the data point $\mathbf{x}$ on directions parallel with the vector formed by $\mathbf{x}$ and its projection $\mathbf{p}$. c) Weights $\mathbf{w}_i$, $\mathbf{w}_j$ and $\mathbf{w}_k$ are displaced towards the data point $\mathbf{x}$ on directions parallel with the vector formed by $\mathbf{x}$ and its projection $\mathbf{p}$. The closer $\mathbf{p}$ is to a node, the larger its update.

$$\begin{aligned}&= \eta_{ji}(\mathbf{x}-\mathbf{w}_i) - 2\eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i) + \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)\\ &= \eta_{ji}(\mathbf{x}-\mathbf{w}_i) - \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)\,. \qquad (16)\end{aligned}$$

Given the above result and with the symmetry in i and j, the two displacements applied to the winner 1D-segment nodes are

$$\Delta\mathbf{w}_{\hat{j}} = \epsilon(t)\eta_{\hat{j}\hat{i}}\left(\mathbf{x}-\mathbf{w}_{\hat{i}}-\eta_{\hat{j}\hat{i}}(\mathbf{w}_{\hat{j}}-\mathbf{w}_{\hat{i}})\right) \qquad (17)$$

$$\Delta\mathbf{w}_{\hat{i}} = \epsilon(t)\eta_{\hat{i}\hat{j}}\left(\mathbf{x}-\mathbf{w}_{\hat{j}}-\eta_{\hat{i}\hat{j}}(\mathbf{w}_{\hat{i}}-\mathbf{w}_{\hat{j}})\right)\,. \qquad (18)$$

The movement of the two nodes is orthogonal to the line segment and illustrated in Fig. 3b.

In case a 2D-segment is closest, like in Fig. 3c, three nodes have to be updated. With gradient descent similar to above we obtain the three displacements

$$\Delta\mathbf{w}_{\hat{i}} = (1-\eta_{\hat{j}\hat{i}}-\eta_{\hat{k}\hat{i}})(\mathbf{x}-\mathbf{w}_{\hat{i}}-\eta_{\hat{j}\hat{i}}\mathbf{d}_{\hat{j}\hat{i}}-\eta_{\hat{k}\hat{i}}\mathbf{d}_{\hat{k}\hat{i}}) \qquad (19)$$

$$\Delta\mathbf{w}_{\hat{j}} = (1-\eta_{\hat{i}\hat{j}}-\eta_{\hat{k}\hat{j}})(\mathbf{x}-\mathbf{w}_{\hat{i}}-\eta_{\hat{j}\hat{i}}\mathbf{d}_{\hat{j}\hat{i}}-\eta_{\hat{k}\hat{i}}\mathbf{d}_{\hat{k}\hat{i}}) \qquad (20)$$

$$\Delta\mathbf{w}_{\hat{k}} = (1-\eta_{\hat{i}\hat{k}}-\eta_{\hat{j}\hat{k}})(\mathbf{x}-\mathbf{w}_{\hat{i}}-\eta_{\hat{j}\hat{i}}\mathbf{d}_{\hat{j}\hat{i}}-\eta_{\hat{k}\hat{i}}\mathbf{d}_{\hat{k}\hat{i}})\,, \qquad (21)$$

this time orthogonal to the triangle.

At this point, a short discussion is required concerning the segment updates. Given that the displacements orthogonal to the line or triangle are of finite size, with each update the respective line or triangle will be slightly enlarged. With many update steps the network might increase over the boarders of the date space. Several solutions to this problem are possible. The most canonical one is to add a "spring-like" "forgetting term" term, which has to be weighted such that an appropriate shortening of the distances of the updated nodes takes place with each update step. Details are explained in [11].

3 Results and Evaluation

Our tracking algorithm uses a Kinect [12] device for data acquisition. The Kinect has an infrared depth sensor and a special microchip that allow obtaining depth information from the scene. The Kinect functions properly for distances in the interval $1.2-3.5$ meters. The image stream is characterized by a 640×480 pixels resolution.

The points corresponding to the hand are extracted with a threshold segmentation performed on the given depth frame, based on the assumption that the hand is the closest object to the camera. This yields the point cloud our algorithm works on (see Fig. 4). Note, that our extended SOM algorithm is applied to each individual frame, each time with 5000 training steps (a training step consists of a random choice of a data point, followed by a competition and update step). The following frame always uses as starting position for the network the result of the previous frame. Only in the very beginning of the tracking procedure we need to initialize the network such that we start with an open hand and the network is aligned to the fingers and the thumb. A random initilization might have problems to converge correctly since the network topology is asymmetrical.

In each new frame, we perform a training during which the network receives as input the data points from the given hand point cloud. Fig. 5 shows how the hand network has converged into the point cloud of a given frame, for the left and right hand. As desired, the network represents the given hand topology by minimizing the mean squared distance between the data points and the network with its 1D and 2D segments.

On the contrary, as illustrated in Fig. 6, by performing a standard SOM learning without using the 1D and 2D segments of the network, the tracker does not manage to represent the given hand topology and converges to an entangled form of the network.

Fig. 7 shows results for different hand postures and bent fingers. In all these cases our extended SOM converged correctly and can now be used for representing corresponding gestures.

4 Conclusion

We presented an extension of the SOM which can successfully be applied to the problem of hand skeleton tracking with 3D cameras. The algorithm is efficient enough to be applied to each individual frame of a kinect camera. The extended SOM algorithm is able to produce a rough, but accurate estimate of the hand pose, at comparatively low computational cost.

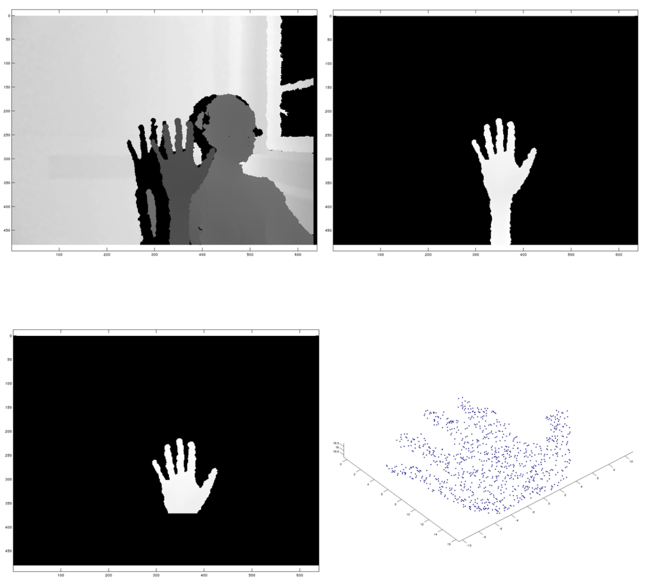

Fig. 4. Starting from a frame captured with the Kinect, we extract the hand based on the closest object assumption. We remove the points corresponding to the forearm and we obtain the 3D point cloud used for the tracker's learning stage.

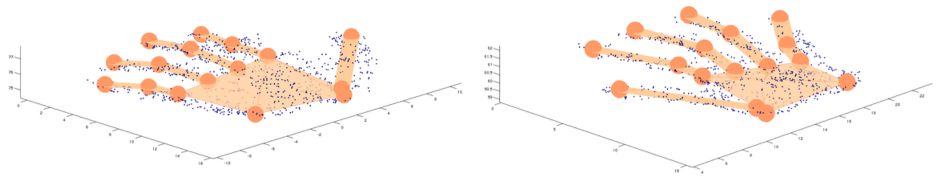

Fig. 5. The extended SOM tracker converges to the open palm topology, for both left and right hand.

Our extension by which we represent the data cloud not only by the nodes of the network, but also by the line and plane segments between the nodes, is necessary for the network to converge correctly to the complicated hand topology. The tracker can be further improved by adding constraints to the hand model, in accordance with the anatomy of the hand.

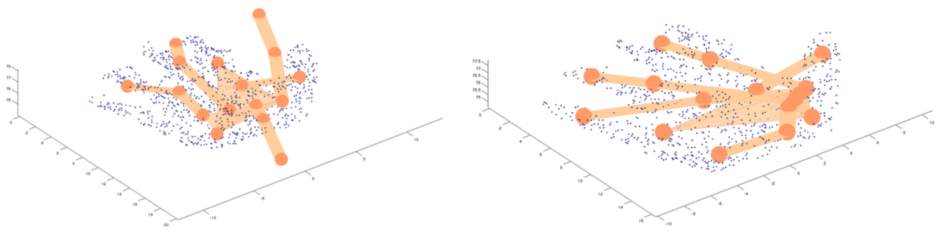
Fig. 6. A standard SOM tracker does not converge to a given hand topology.

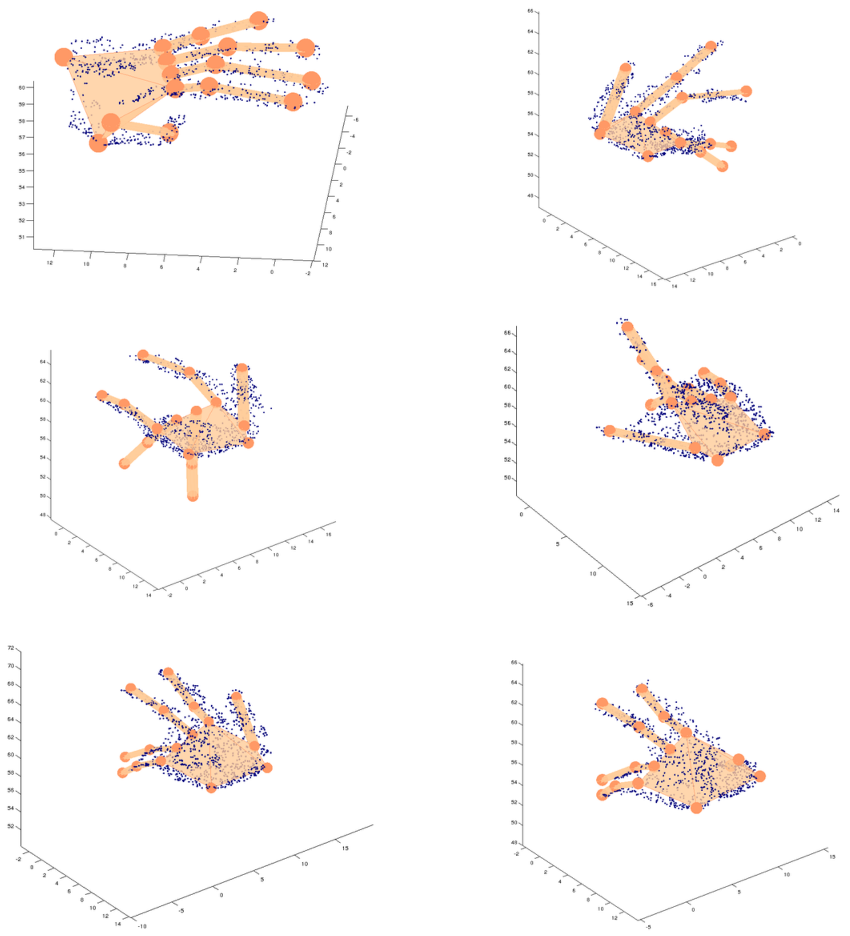
Fig. 7. Different hand postures and bent fingers. The extended SOM converges to the correct topology.

Our method can easily be applied not only to hand tracking, but also to other types of objects, e.g., the human body or animals, by simply varying the SOM network topology. In further developments we will use the network topology to infer useful features for gesture recognition.

Acknowledgments: This research is supported by the German Ministry of Education and Research (BMBF) under grant number 01IS10049B. A. State would like to thank for the support by The German Academic Exchange Service Programme "Ostpartnerschaften", the University of Lübeck and The Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/86/1.2/S/61756.

References

1. ElKoura, G., Singh, K.: Handrix: Animating the Human Hand. In: Proc. SCA, 110–119 (2003)
2. Haker, M., Böhme, M., Martinetz, T., Barth, E.: Deictic gestures with a time-of-flight camera. In: Gesture in Embodied Communication and Human-Computer Interaction – International Gesture Workshop (2009)
3. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (1995)
4. Rehg, J., Kanade, T.: Visual tracking of high DOF articulated structures: An application to human hand tracking. In: Proc. 3rd European Conference on Computer Vision, J.-O. Eklundh (Ed.), Springer-Verlag, pp. 35–46. (1994)
5. Segen, J., Kumar, S.: Shadow Gestures: 3D Hand Pose Estimation using a Single Camera". In: Proceedings of Conference on Computer Vision and Pattern Recognition (1999)
6. Rosales, R., Athitsos, V., Sclaroff, S.: 3D hand pose reconstruction using specialized mappings. In: Proc. International Conference on Computer Vision, volume 1, pp 378–385 (2001)
7. Athitsos, V., Sclaroff, S.: Estimating 3D hand pose from a cluttered image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2003)
8. Stenger, B., Thayananthan, A., Torr, P. H. S., Cipolla, R.: Hand Pose Estimation Using Hierarchical Detection. In: Proc. International Workshop Human-Computer Interaction, pp. 105–116 (2004)
9. Bray, M., Koller-Meier, E., Gool, L. V.: Smart particle filtering for 3D hand tracking. In: Sixth IEEE International Conference on Automatic Face and Gesture Recognition, IEEE Computer Society, Los Alamitos, CA, USA, p. 675 (2004)
10. Haker, M., Böhme, M., Martinetz, T., Barth, E.: Self-organizing maps for pose estimation with a time-of-flight camera. In: Proceedings of the DAGM Workshop on Dynamic 3D Imaging – Lecture Notes in Computer Science Volume 5742, pp 142–153 (2009)
11. Ehlers, K., Timm, F., Barth, E., Martinetz, T.: A generalization of k-means for real time hand skeleton tracking. In preparation.
12. Kinect home page, `http://www.xbox.com/en-US/kinect/`