Proc. Int. Conf. on Robotics and Automation (ICRA) IEEE 1998

PSOM Network: Learning with Few Examples

Jörg A. Walter

Department of Computer Science · University of Bielefeld · D-33615 Bielefeld
Email: walter@techfak.uni-bielefeld.de

Abstract:

Precise sensorimotor mappings between various motor, joint, sensor, and abstract physical spaces are the basis for many robotics tasks. Their cheap construction is a challenge for adaptive and learning methods. However, the practical application of many neural networks suffer from the need of large amounts of training data, which makes the learning phase a costly operation – sometimes beyond reasonable bounds of cost and effort.

In this paper we discuss the "Parameterized Self-organizing Maps" (PSOM) as a learning method for rapidly creating high-dimensional, continuous mappings. By making use of available topological information the PSOM shows excellent generalization capabilities from a small set of training data. Unlike most other existing approaches that are limited to the representation of a input-output mappings, the PSOM provides as an important generalization a flexibly usable, *continuous associate memory*. This allows to represent several related mappings – coexisting in a single and coherent framework.

Task specifications for redundant manipulators often leave the problem of picking one action from a subspace of possible alternatives. The PSOM approach offers a flexible and compact form to select from various constraint and target functions previously associated.

We present application results for learning several kinematic relations of a hydraulic robot finger in a single PSOM module. Based on only 27 data points, the PSOM learns the inverse kinematic with a mean positioning accuracy of 1 % of the entire workspace. Another PSOM learns various ways to resolve the redundancy problem for positioning a 4 DOF manipulator.

1 Introduction

Many tasks in robotics require the availability of precise sensorimotor mappings – able to transform between various motor, joint, sensor, and abstract physical spaces. The construction of required relationship from empirical training data is a challenge for adaptive and learning methods. Unfortunately, many neural network approaches indeed require hundreds or thousands of examples and training steps. Since the acquisition of this data is related to cost and effort, this is a major obstacle for the practical application of those methods.

To make a learning system useful and efficient in robotics means that the learner provides *good generalization* capabilities – based on a *small* training data set, and uses a quick learning procedure without fragile learning parameters and without taking too much iteration time (with growing availability of computing power this need is increasingly relaxed, allowing also more elaborated algorithms to compete).

In this contribution we present the "Parameterized Self-Organizing Map" (PSOM) approach, which is particularly useful in situation where a high-dimensional, continuous mapping is desired. If information about the topological order of the training data is provided, or can be inferred, only a very small data set is required. In section 2 the PSOM algorithm is derived from Kohonen's Self-Organizing Map and the PSOM's *auto-associative* capabilities are presented.

In section 3 we report on a PSOM application for solving the forward and backward kinematics for a robot finger. If numerous degrees of freedom are available one has to pick a configuration of a continuous space of alternatives. Most solutions of this redundancy problem are based on some pseudo-inverse control (for a review see e.g. [2]). However a more flexible solution should offer a set of suitable action strategies and should offer to respond to different types of constraints. Sec. 4 suggests an *associative memory* that completes partial specification of a incomplete task specification as a natural solution. But in contrast to spin-glass type attractor networks, in robotics, we need a continuous attractor manifold instead of just isolated points. Here we show that a PSOM can provide the functionality of representing continuous relations in conjunction with a favorable flexibility to specify additional goals.

2 From SOMs to PSOMs

Kohonen [3] formulated the *Self-Organizing Map* (SOM) algorithm as a mathematical model of the self-organization of topographic maps, which are found in brains of higher animals. Fig. 1 illustrates a two-dimensional array A of processing units or formal "neurons". Each neuron has a reference vector $\mathbf{w_a}$ attached, which points in the embedding input space X. A presented input $\mathbf{x}$ will select that neuron $\mathbf{a}^*$ with $\mathbf{w_a}$ closest to the given input: $\mathbf{a}^* = \text{argmin}_{\forall \mathbf{a} \in \mathbf{A}} \|\mathbf{w_{a'}} - \mathbf{x}\|$. This competitive mechanism tessellates the input space in *discrete* patches – the so-called *Voronoi cells* (see light gray border lines).

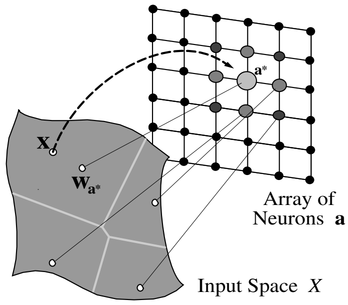

Figure 1: The "Self-Organizing Map" ("SOM") algorithm builds a topographic map and tessellates the input space to discrete Voronoi-cells.

The Kohonen learning rule (see e.g. [3, 6]) generates a dimension reducing, topographic mapping from a high-dimensional input space to a m-dimensional index space of neurons in the array S. Topographic order means that neighboring neurons are responsible for similar input situations $\in X$. The main features are: *(i)* the generation of the *topographic order* and, *(ii)* as a result, each learning step can profitably be shared among neighboring neurons, allowing to improve the convergence properties of the algorithm.

How can the SOM-network learn a smooth continuous input–output mapping? A simple strategy is the supervised teaching of a constant output value $y_\mathbf{a}$ (or vector $\mathbf{y_a}$) per neuron $\mathbf{a}$. The network output is then $F(\mathbf{x}) = y_{\mathbf{a}^*}$ of the winner neuron $\mathbf{a}^*$. The first improvement to increase the output precision is the introduction of a locally linear regression scheme for each neuron $\mathbf{a}$ and returning the "winner" neuron's output: $F(\mathbf{x}) = y_{\mathbf{a}^*} + \mathbf{B}_{\mathbf{a}^*}(\mathbf{x} - \mathbf{w}_{\mathbf{a}^*})$; i.e. a set of (hyper-) planes approximate the desired function. Unfortunately, in general the planes do not match at the borders of the Voronoi-cells, which may leave discontinuities in the overall mapping.

The PSOM concept [5] can be seen as the generalization of the SOM with the following three main extensions:

- the index space S in the Kohonen map is generalized to a *continuous mapping manifold* $S \in \mathbb{R}^m$.
- The embedding space $X = X^{in} \times X^{out} \subset \mathbb{R}^d$ is formed by the Cartesian product of the input space and output space.
- We define a *continuous mapping* $\mathbf{w}(\cdot) : \mathbf{s} \mapsto \mathbf{w}(\mathbf{s}) \in M \subset X$, where $\mathbf{s}$ varies continuously over $S \subseteq \mathbb{R}^m$.

We require that the *embedded manifold* M passes through all supporting reference vectors $\mathbf{w_a}$ and write $\mathbf{w}(\cdot) : S \rightarrow M \subset X$ as weighted sum:

$$\mathbf{w}(\mathbf{s}) = \sum_{\mathbf{a} \in \mathbf{A}} H_\mathbf{a}(\mathbf{s})\, \mathbf{w_a} \tag{1}$$

This means that, we need a *"basis function"* $H_\mathbf{a}(\mathbf{s})$ for each formal neuron or "node", weighting the contribution of its reference vector (= initial "training point") $\mathbf{w_a}$. The $H_\mathbf{a}(\mathbf{s})$ depend on the location $\mathbf{s}$ relative to the node position $\mathbf{a}$, and also *all* other nodes $\mathbf{A}$ (however, we drop in our notation the dependency $H_\mathbf{a}(\mathbf{s}) = H_{\mathbf{a};\mathbf{A}}(\mathbf{s})$ on $\mathbf{A}$).

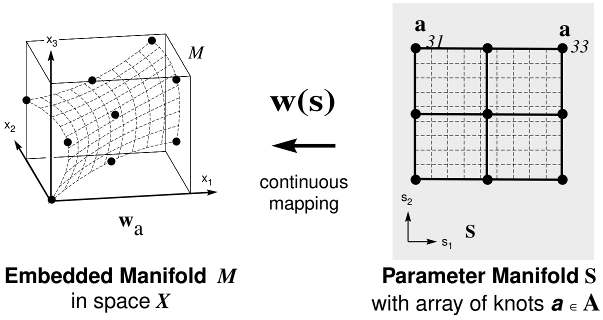

Figure 2: The mapping $\mathbf{w}(\cdot) : S \rightarrow M \subset X$ builds a continuous image of the *right* side S in the embedding space X, as illustrated by the dotted test grid.

A suitable set of basis functions can be constructed in several ways but must meet two conditions: *(i)* the hypersurface M shall pass through all desired support points (*orthonormality*), i.e. at those points, only the local node contributes $H_{\mathbf{a}_i}(\mathbf{a}_j) = \delta_{ij}$; $\forall\, \mathbf{a}_i, \mathbf{a}_j \in \mathbf{A}$; *(ii)* the sum of all contribution weights must be one: $\sum_{\mathbf{a} \in \mathbf{A}} H_\mathbf{a}(\mathbf{s}) = 1,\ \forall \mathbf{s}$ *(partition-of-unity)*.

A simple construction of basis functions $H_\mathbf{a}(\mathbf{s})$ becomes possible when the topology of the given points is sufficiently regular. A particularly convenient situation arises for the case of a multidimensional rectangular grid. In this case, the set of functions $H_\mathbf{a}(\mathbf{s})$ can be constructed from products of one-dimensional Lagrange interpolation polynomials. See [7] for details.

Specifying for each training vector $\mathbf{w_a}$ a node location $\mathbf{a} \in \mathbf{A}$ introduces a *topological order* between the training points: training vectors assigned to nodes $\mathbf{a}$ and $\mathbf{a}'$, that are adjacent in the lattice $\mathbf{A}$, are perceived to have this specific neighborhood relation. The effect is important to note: it allows the PSOM to *draw extra curvature information* from the training set. Such information is not available within other techniques, such as the RBF approach and is the essential reason for the generalization capabilities of the PSOM (see below Fig. 5–7).

When M has been specified, the PSOM is used similar to the SOM: given an input vector $\mathbf{x}$, *(i)* find the best-match position $\mathbf{s}^*$ on the mapping manifold S by minimizing the distance function $dist(\cdot)$

$$\mathbf{s}^* = \mathbf{s}(\mathbf{x}) = \underset{\forall \mathbf{s} \in S}{\operatorname{argmin}} \; dist\left(\mathbf{w}(\mathbf{s}), \mathbf{x}\right). \qquad (2)$$

(ii) The surface point $\mathbf{w}(\mathbf{s}^*)$ serves as the output of the PSOM in response to the input $\mathbf{x}$. The output $\mathbf{w}(\mathbf{s}^*)$ can be viewed as an *associative completion* of the input space component of $\mathbf{x}$ if the distance function $dist(\cdot)$ (in Eq. 2) is chosen as the Euclidean norm applied only to the input components of $\mathbf{x}$ (belonging to X^{in}). Thus, the function $dist(\cdot)$ actually selects the input subspace X^{in}, since for the determination of $\mathbf{s}^*$ (Eq. 2) and, as a consequence, of $\mathbf{w}(\mathbf{s}^*)$, only those components of $\mathbf{x}$ matter, that are regarded in the distance metric $dist(\cdot)$. A suitable definition is

$$dist(\mathbf{x}, \mathbf{x}') = \sum_{k=1}^{d} p_k \, (x_k - x'_k)^2. \qquad (3)$$

which selects all components k with $p_k > 0$ as belonging to the input subspace; output are components k with $p_k = 0$. By changing the coefficients p_μ the PSOM the mapping direction can be (e.g.) reversed, as illustrated in Fig. 3.

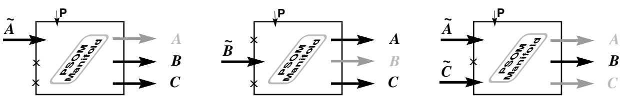

Figure 3: "Continuous associative memory" supports multiple mapping directions. The specified $\mathbf{P}$ vector selects different subspaces (here symbolized by $\tilde{A}$, $\tilde{B}$ and $\tilde{C}$) of the embedding space as inputs. Values of variables in the selected input subspaces are considered as "clamped" (indicated by a tilde) and determine the values found by the iterative least square minimization (Eq. 2). for the "best-match" vector $\mathbf{w}(\mathbf{s}^*)$. This provides an associative memory for the flexible representation of continuous relations.

The discrete best-match search in the standard SOM is now replaced by solving the continuous minimization problem for the determination of $\mathbf{s}^*$ in Eq. 2. A simple approach is to first perform the (SOM-like) discrete best-match search to find $\mathbf{s}_{start} = \mathbf{a}^*$ in the knot set $\mathbf{A}$, followed by an iterative procedure like the gradient descent. We found the Levenberg-Marquardt algorithm [4] best suited to find $\mathbf{s}^*$ in a couple of iterations.

In this scheme M can be viewed as a continuous attractor manifold with a recurrent dynamic. Since M contains the data set $\{\mathbf{w_a}\}$, any at least m-dimensional "fragment" of the data set will be attracted to the completion $\mathbf{w}$. Any other input will be attracted to an interpolation manifold point.

3 Application: The Robot Finger Kinematics

This section presents the results of applying the PSOM algorithm to the task of learning the kinematics of a 3 degree-of-freedom robot finger of a three-fingered modular hydraulic robot hand, developed by the Technical University of Munich. The finger is actuated by spring-loaded oil cylinders driven by a remote "base station" that provides the hydraulic pressure. Its mechanical design allows roughly the mobility of the human index finger, scaled up to 110%. A cardanic base joint (2 DOF) offers sidewards gyring of $\pm 15°$ and full adduction with two additional coupled joints (1 DOF). See Fig. 4.

In the case of the finger, there are several coordinate systems of interest, e.g. the joint angles $\vec{\theta}$, the cylinder piston positions $\vec{c}$, one or more finger tip coordinates $\vec{r}$, as well as further configuration dependent quantities, such as the Jacobian matrices J for force/moment transformations. All of these quantities can be simultaneously treated in one single PSOM allowing to map in multiple ways, as indicated in Fig. 3. Here we present results of the inverse kinematics, the classical hard part. When moving the three joints on a cubical $10 \times 10 \times 10$ grid within their maximal configuration space, the fingertip will trace out the "banana" grid displayed in Fig. 4 (confirm this workspace with your finger).

We exercised several PSOMs with n×n×n nine dimensional data tuples $(\vec{\theta}, \vec{c}, \vec{r})$, all equidistantly sampled in $\vec{\theta}$. Fig. 5a–b depicts a $\vec{\theta}$ and an $\vec{r}$ projection of the smallest training set, $n = 3$.

To visualize the inverse kinematics ability, we ask the PSOM to back-transform a set of workspace points of known arrangement. In particular, the workspace filling "banana" set of Fig. 4 should yield a rectangular grid of $\vec{\theta}$. Fig. 5c–e displays the actual result. Distortions can be vi-

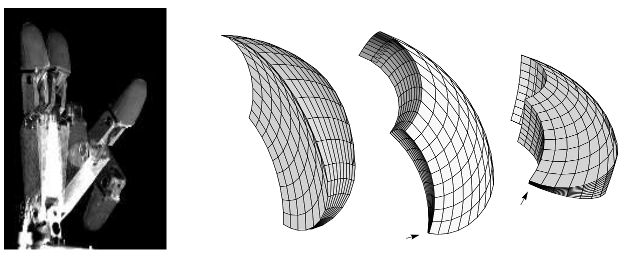

Figure 4: a–d: *(a)* stroboscopic image of one finger in a sequence of extreme joint positions. *(b–d)* Several perspectives of the workspace envelope $\vec{r}$, tracing out a cubical $10 \times 10 \times 10$ grid in the joint space $\vec{\theta}$. The arrow marks the fully adducted position, where one edge contracts to a tiny line.

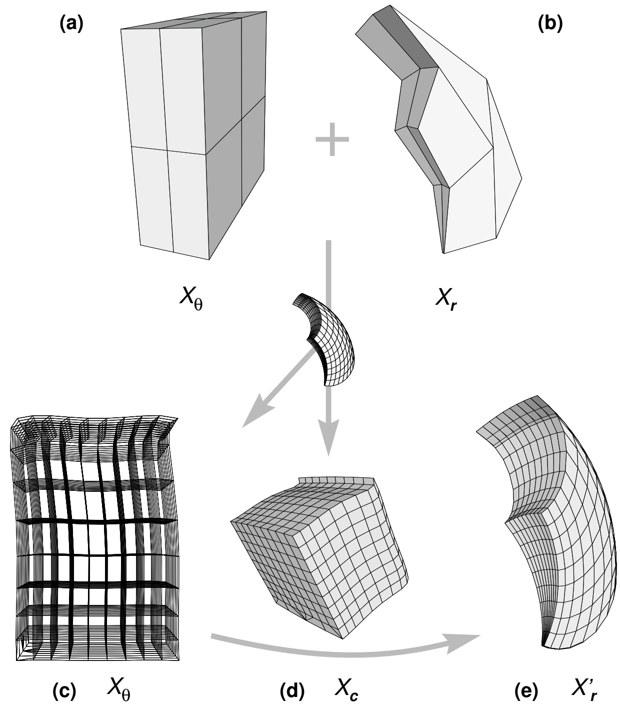

Figure 5: a–b and c–e; Training data set of 27 nine-dimensional points in X for the $3{\times}3{\times}3$ PSOM, shown as perspective surface projections of the *(a)* joint angle $\vec{\theta}$ and *(b)* the corresponding Cartesian sub space. Following the lines connecting the training samples allows one to verify that the "banana" really possesses a cubical topology. *(c–e)* Inverse kinematic result using the grid test set displayed in Fig. 4. *(c)* projection of the joint angle space $\vec{\theta}$ (transparent); *(d)* the stroke position space $\vec{c}$; *(e)* the Cartesian space $\vec{r}\,'$, after back-transformation.

sually detected in the joint angle space *(c)*, and the piston stoke space *(d)*, but disappear after back-transforming the PSOM output to world coordinates *(b)*. The reason is the peculiar structure; e.g. in areas close to the tip a certain angle error corresponds to a smaller Cartesian deviation than in other areas.

When measuring the mean Cartesian deviation we get an already satisfying result of 1.6 mm or **1.0 %** of the maximum workspace length of 160 mm (using a test set of 500 randomly chosen positions). In view of the extremely small training set displayed in Fig. 5a–b this appears to be a quite remarkable result.

Nevertheless the result can be further improved by supplying more training points. For a growing number of network nodes the "Local-PSOM" approach offers to keep the computational effort constant by applying the PSOM algorithm on a sub-grid, see [8, 7].

PSOM versus MLP: For comparison reasons, we employed the standard Multi-Layer-Perceptron with one and two hidden layers and linear units in the output layer. We found that this problem is not suitable for the MLP network. Even for larger training set sizes, we did not succeed in training them to a performance comparable to the PSOM network.

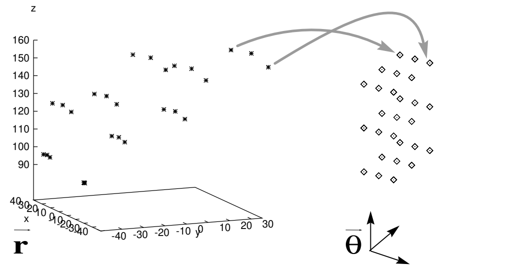

Figure 6: The 27 training data vectors for the Back-propagation networks: *(left)* in the input space $\vec{r}$ and *(right)* the corresponding target output values $\vec{\theta}$.

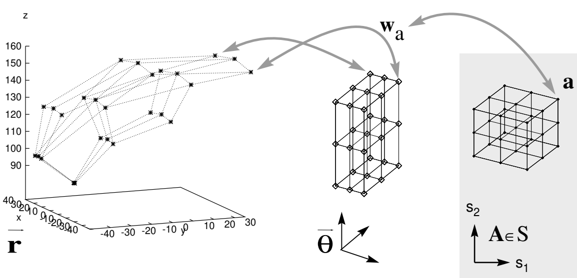

Figure 7: The same 27 training data vectors (cmp. Fig. 6) for the bi-directional PSOM mapping: *(left)* in the Cartesian space $\vec{r}$, *(middle)* the corresponding joint angle space $\vec{\theta}$. *(Right:)* The corresponding node locations $\mathbf{a} \in \mathbf{A}$ in the parameter manifold S. Neighboring nodes are connected by lines, which reveals now the "banana" structure on the left.

Why does the PSOM perform more than an order of magnitude better than the back-propagation algorithm? Fig. 6 shows the 27 training data pairs; on the left side, the Cartesian input space $\vec{r}$, one can recognize some zigzag structure, but not much more. Fig. 7 depicts the PSOM situation: the PSOM gets the same data-pairs as training vectors — but additionally, it obtains the assignment to the node location $\mathbf{a}$ in the $3{\times}3{\times}3$ node grid illustrated in Fig. 7. If neighboring nodes are connected by lines, it is easy to recognize the coarse "banana" shaped structure. Using the curvature information contained in the ordered data the PSOM could generalized to the reported positioning precision of 1%. This topological information is not available to other techniques, like the MLP or the radial basis approach.

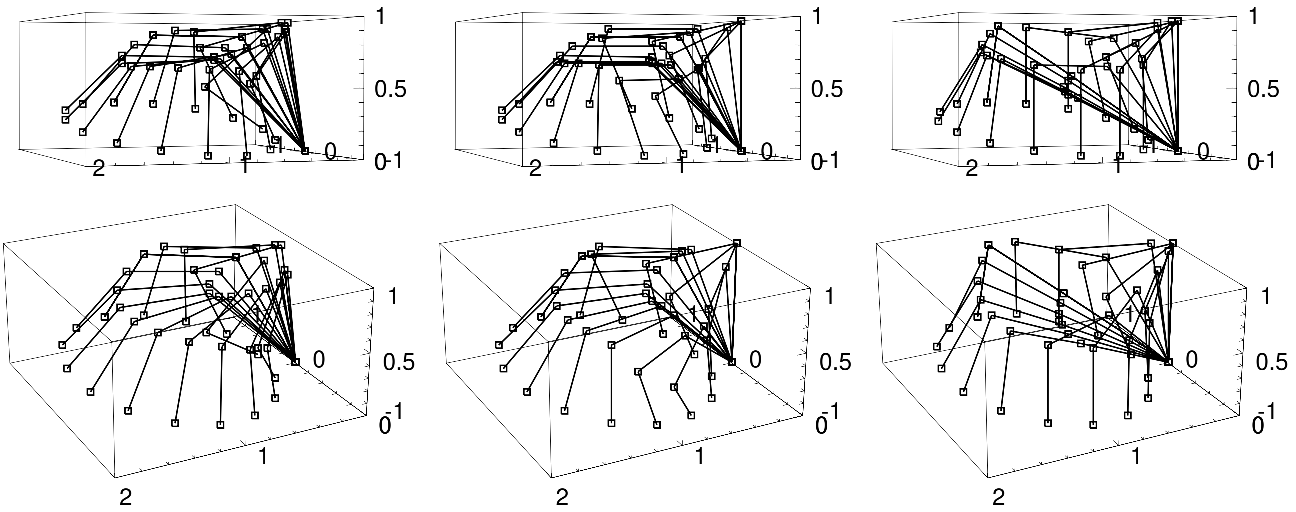

Figure 8: Tracking a point in a slanted elliptical path with a 4 DOF manipulator, using a 5×5×5×5 $PSOM$ and three different extra target functions to resolve the redundancy problem: *(left)* with maximal similarity of the last two joints; *(middle)* with horizontal middle arm segment; *(rigth)* with vertical distal arm segment – as far as possible.

4 Application: Flexible Use of Redundant DOF

As mentioned earlier, in the presence of excess degrees of freedom one may specify extra constraints to determine a robot configuration. The question is how this can be done in a versatile manner? Here, the PSOM contributes an elegant way to offer several goal and contraint functions for flexible usage.

We illustrate the flexibility of the PSOM approach in the task to position a 4 DOF robot in 3D as depicted in Fig. 8. The simulated robot consists of a vertical revolute joint θ_1 and three horizontal ones $\theta_{2..4}$ with link lengths 0,1,0.7,0.6 l. For the construction of a PSOM, i.e. the configuration manifold of the arm, we choose an 5×5×5×5 grid covering the joint range $\vec{\theta} \in [0°, 180°] \times [0°, 120°] \times [-120°, 0°] \times [-120°, 0°]$. The embedding space X is spanned by $\mathbf{x} = (\theta_1, \theta_2, \theta_3, \theta_4, r_x, r_y, r_z, c_8, c_9, c_{10})^T$ and contains, similar to the example before, the angles $\theta_{1..4}$, the Cartesian position $\vec{r}$, and, here, three further parameters: c_8 is the difference $(\theta_4 - \theta_3)^2$, c_9 is the elevation angle of link-3 (relative to the horizontal), and c_{10} is the angle between distal link-4 and the vertical.

This allows to resolve the redundancy in various ways. For example, the goal can be to ...

(i) take the minimal joint motion from the current position to the specified position $\vec{r}$: all we need to do is to start the best-match search ($p_{5..7} = 1$) at the best-match position $\mathbf{s}^*_{curr}$ belonging to the current position, and the steepest gradient descent procedure will solve the problem;

(ii) keep joint $j \in \{2, 3, 4\}$ fixed: additionally specify θ_j and $p_j > 0$;

(iii) use similar adduction in the two distal joints (like the finger kinematics): by activating $p_8 > 0$ (to a small value e.g., 0.01) and setting $x_9 = 0$. Measuring the deviation for the inverse kinematics we find a mean value of 0.008 in the workspace; Fig. 8*(left)* depicts the solution for tracing a elliptical path for the end effector;

(iv) keep the middle segment horizontal: by specifying the target $x_9 = 0$ and $p_9 = 0.01$. Fig. 8*(middle)* reveals that this constraint can not be met in all cases. By setting p_9 to only a small value, as a "soft goal", the accuracy of the trajectory is not (significantly) compromised (see also below);

(v) approach vertically: after specifying $x_{10} = 0$, $p_{10} = 0.01$, Fig. 8*(right)* shows the stroboscopic tracking result.

For these different cases we do not need different networks, instead one single PSOM can be utilized. If one anticipates useful target functions, the embedding space can be augmented in advance, enabling to construct reconfigurable optimization modules. They are later activated on demand and show the desired performance. In conflicting situation, e.g. the distal reaching positions in the last example, a meaningful compromise is found. As shown in [7], the input selection coefficients can be made dynamical $p_\mu = p_\mu(t)$ during the iteration process. I.e., secondary goal functions are weighted by $p_\mu(t)$, starting at a small value, which decrease to zero. Primary positioning goals are not compromised and secondary goals satisfied as much as possible. This procedure allows to define priorities of goal functions, which are solved according to there rank.

5 Discussion and Conclusion

We presented the PSOM as a versatile module for learning continuous, high-dimensional mappings. As highlighted by the robot finger example, the PSOM draws its good generalization capabilities from curvature information available through the topological order of only a few reference vectors $\{\mathbf{w_a}\}$. This topological assignment can be learned by *(i)* Kohonen's SOM learning rule, or *(ii)* by construction – if the topological relation of the data is known. The first is an iterative learning method which requires more training data than the second. We find that in many robotic applications, the latter case can be realized by active, structured sampling of the training data – often without any extra cost. This can be viewed as a special mechanism for incorporating prior knowledge into the learning system. In the robot finger example a set of only 27 data points turns out sufficient to approximate the highly non-linear 3 D kinematics relation with remarkable precision. The inverse kinematics showed a mean positioning deviation of only 1 % of the entire workspace range.

Due to the compactness of the training set, the PSOM has some overlap with fuzzy networks (e.g. [10]): An expert defines a fuzzy class, assigns linguistic names (e.g. "left", "middle", "right stroke position"), and (initially) provides suitable output values. In the PSOM learning process, the grid node values $\mathbf{a}$ can be assigned to input-output pairs. Likewise, names can be alloted to support the interpretation of the learned knowledge.

The PSOM's associative mapping concept has various attractive properties. Several coordinate spaces can be maintained and learned simultaneously, as shown in the robot finger example. E.g., the *multi-way mapping* capability can solve the forward and inverse kinematics within the very same network. This simplifies learning and avoids worries about inconsistencies of separate learning modules. As pointed out by Kawato [1], the learning of bi-directional mappings is not only useful for the planning phase (action simulation), but also for bi-directional sensor–motor integrated control.

Another potential PSOM application is the representation of system states together with a set of values from different sensors. Here, the PSOM can serve as an integrated sensor data fusion mechanism. It allows not only to incrementally fuse available data, but also to deliver intermediate data predictions, usable for sensor guidance. For details we must refer to [7] where also examples of the full 6 DOF robot kinematics can be found together with responses to sudden changes in underlying mapping task. See [9] for several applications in visuo-motor coordination.

The input selection mechanism enables to easily add further, parameterized target functions. Those can be utilized to resolve redundancy problems when tasks are underspecified. Here the PSOM offers to build a battery of optimizer modules which can be learned within the same continuous associative memory. On demand, one can select or combine auxiliary goals by activating the components in the distance metric (p_k in Eq. 3). The continuous associative completion serves here as an elegant and compact mechanism to provide a variety of options and solutions.

References

[1] Mitsuo Kawato. Bi-directional neural network architecture in brain functions. In *Proc. Int. Conf. on Artificial Neural Networks (ICANN-95), Paris*, volume 1, pages 23–30, 1995.

[2] C.A. Klein and C.-H. Huang. Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Trans. Sys. Man and Cybern.*, 13:245–250, 1983.

[3] Teuvo Kohonen. *Self-Organization and Associative Memory*. Springer Series in Information Sciences 8. Springer, Heidelberg, 1984.

[4] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C – the Art of Scientific Computing*. Cambridge Univ. Press, 1988.

[5] Helge Ritter. Parametrized self-organizing maps. In S. Gielen and B. Kappen, editors, *Proc. Int. Conf. on Artificial Neural Networks (ICANN-93), Amsterdam*, pages 568–575. Springer Verlag, Berlin, 1993.

[6] Helge Ritter, Thomas Martinetz, and Klaus Schulten. *Neural Computation and Self-organizing Maps*. Addison Wesley, 1992.

[7] Jörg Walter. *Rapid Learning in Robotics*. Cuvillier Verlag Göttingen, 1996. also postscript http://www.techfak.uni-bielefeld.de/∼walter/pub/.

[8] Jörg Walter and Helge Ritter. Local PSOMs and Chebyshev PSOMs – improving the parametrised self-organizing maps. In *Proc. Int. Conf. on Artificial Neural Networks (ICANN-95), Paris*, volume 1, pages 95–102, 1995.

[9] Jörg Walter and Helge Ritter. Rapid learning with parametrized self-organizing maps. *Neurocomputing*, 12:131–153, 1996.

[10] J. Zhang, Y. von Collani, and A. Knoll. On-line learning of b-spline fuzzy controller to acquire sensor-based assembly skills. In *Proc. Int. Conf. on Robotics and Automation (ICRA-97)*, pages 1418–1423, 1997.