Proc. Int. ICSC Symposium on Neural Computertion (NC'2000) (in press)

The PSOM Algorithm and Applications

Jörg Walter · Claudia Nölker · Helge Ritter

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

Abstract:

In this paper we discuss the "Parameterized Self-organizing Maps" (PSOM) as a learning method for rapidly creating high-dimensional, continuous mappings. The PSOM can be viewed as the continous generalization of the discrete topology preserving map build by Kohonen's SOM algorithm. 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 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.

We present application examples for simultaneous learning of robot forward and backward kinematics, concepts for an integrated redundancy control scheme, and the application to gesture recognition in a human-machine-interface. All profit from the approximation accuracy gained from only a few training examples.

1 Introduction

Many learning tasks can be formulated as mapping process. I.e. robotics requires the availability of precise sensorimotor mappings – able to transform between various motor, joint, sensor, and abstract physical spaces. The construction of required relationships 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 means that *(i)* the learner exhibits *good generalization* capabilities; *(ii)* it can benefit already from a *small* training data set, and *(iii)* it uses a quick learning procedure without fragile learning parameters and without taking too much iteration time (the growing computing power enables here more and more elaborated algorithms to compete).

When a high-dimensional, continuous mapping is desired, the PSOM might be a useful candidate. In particular, 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 3.1 we report on a PSOM application for solving the forward and backward kinematics for a robot finger. If redundant degrees of freedom are available one has to pick a configuration from 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. 3.2 suggests an *associative memory* that completes partial 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.

Furthermore, we present *GREFIT*, a system for *G*esture *RE*cognition based on *FI*nger *T*ips seen in video images. Watching the unadorned human hand by a camera, the system can determine the hand posture in a continuous parametrization. The shape reconstruction can be conveniently inspected by displaying a hand model, articulated with 20 degrees of freedom.

2 From SOMs to PSOMs

Teuvo 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. The SOM consists (usually) of a two-dimensional array $\mathbf{A}$ 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: $\mathbf{a}^* = \operatorname{argmin}_{\forall \mathbf{a} \in \mathbf{A}} \|\mathbf{w}_{\mathbf{a}'} - \mathbf{x}\|$. This competitive mechanism tessellates the input space into *discrete* patches – the so-called *Voronoi cells*.

The Kohonen learning rule (see e.g. [3]) generates a dimension reducing, topographic mapping from a high-dimensional input space to a m-dimensional index space of neurons in the array $\mathbf{A}$. Topographic order means that neighboring neurons are responsible for similar input situations ($\in X$).

How can the SOM-network learn an output, or better a continuous input–output mapping? The simplest strategy is the supervised teaching of a constant output value $y_{\mathbf{a}}$ (or vector $\mathbf{y}_{\mathbf{a}}$) 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 *l*ocally valid *l*inear *m*ap (LLM), a local regression scheme for each neuron $\mathbf{a}$. The "winner" neuron response is then given by $F(\mathbf{x}) = y_{\mathbf{a}^*} + \mathbf{B}_{\mathbf{a}^*}(\mathbf{x} - \mathbf{w}_{\mathbf{a}^*})$; i.e. a set of (hyper-) planes approximates the desired function. Unfortunately, in general the planes do not match at the borders of the Voronoi-cells, which will leave discontinuities in the overall mapping.

The PSOM concept [6] generalizes the SOM in the following three main points:

- the index space S in the Kohonen map is generalized to a *continuous mapping manifold* $S \in \mathbb{R}^m$.
- The embedding space $X = X^{in} \otimes X^{out} \subset \mathbb{R}^d$ 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 $S \subseteq \mathbb{R}^m$.

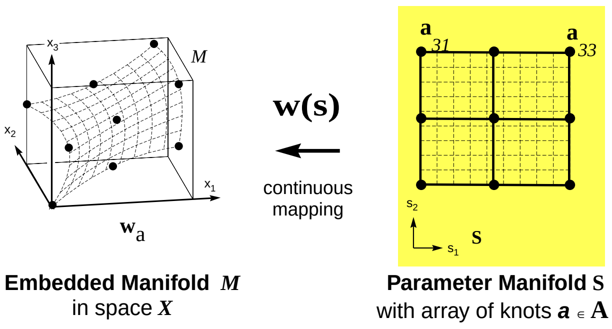
The latter defines an *embedded manifold* M which we require to pass through all supporting reference vectors $\mathbf{w}_{\mathbf{a}}$ and write $\mathbf{w}(\cdot) : S \to M \subset X$ as weighted sum:

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

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") $\mathbf{w}_{\mathbf{a}}$. The $H_{\mathbf{a}}(\mathbf{s})$ depend on the location $\mathbf{s}$ relative to the node position $\mathbf{a}$, and also on *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 $\mathbf{A}$).

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



x3
M
x2
x1
w_a
w(s)
continuous mapping
a 31
a 33
s2
s1
S
Embedded Manifold M in space X
Parameter Manifold S with array of knots a ∈ A

Figure 1: The **continuous mapping** $\mathbf{w}(\cdot) : S \to M \subset X$ builds a image of the *right* side S in the embedding space X, as illustrated by the dotted test grid.

polynomials. Fig. 2 depicts three (of nine) basis functions $H_{\mathbf{a}}(\mathbf{s})$ for the $m = 2$ dimensional example with a 3×3 rectangular node grid $\mathbf{A}$ shown in Fig. 1.

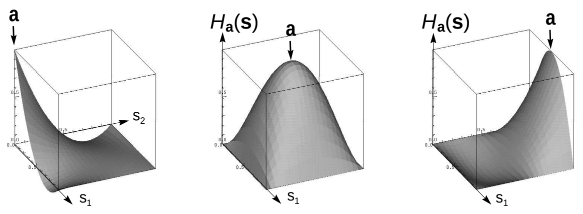


Figure 2: Three of the nine **basis functions** $H_{\mathbf{a}}(\cdot)$ for a 3×3 PSOM with equidistant node spacing $\mathbf{A} = \{0, \frac{1}{2}, 1\} \otimes \{0, \frac{1}{2}, 1\}$. The remaining six basis functions are obtained by $90°$ rotations.

Specifying for each training vector $\mathbf{w}_{\mathbf{a}}$ 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: it allows the PSOM to *draw extra curvature information* from the training set. As we explore later, this is the essential reason for the favorable generalization capabilities of the PSOM (see below Fig. 7–9).

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

$$\mathbf{s}^* = \mathbf{s}(\mathbf{x}) = \underset{\forall \mathbf{s}\in S}{\text{argmin}}\ dist\,(\mathbf{w}(\mathbf{s}), \mathbf{x})\,. \tag{2}$$

(ii) The output of the PSOM in response to the input $\mathbf{x}$ is the surface point $\mathbf{w}(\mathbf{s}^*)$. This 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) depends only on the input components of $\mathbf{x}$ (belonging to X^{in}). Or, in other words, the function $dist(\cdot)$

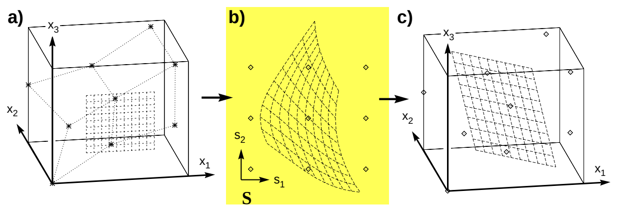


Figure 3: A (trivial) **mapping example** of Eq. 4: *(a)* The 3×3 training points lay on a slanted plane (=Fig. 1); the test set is the grid in front. *(b)* s^* set resulting from Eq. 2. *(c)* The image in X lays on the $\{\mathbf{w_a}\}$-defi ned plane. This (non-demanding) task shows the ability of *associative-completion* and the non-linear mapping behavior – back and forth.

actually selects the input subspace X^{in}: since for the determination of $\mathbf{s}^*$ (Eq. 2) and, as a consequence, for $\mathbf{w}(\mathbf{s}^*)$, only those components of $\mathbf{x}$ matter, that are regarded in the distance metric $dist(\cdot)$. A suitable definition is

$$dist(\mathbf{x},\mathbf{x}') = \sum_{k=1}^{d} p_k\,(x_k - x'_k)^2\ ,\ \ p_k \geq 0. \qquad (3)$$

which selects all components k with non-zero p_k as belonging to the input subspace; output are components k with $p_k = 0$. By changing the coefficients p_μ the PSOM mapping direction can be (e.g.) reversed on demand.

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 *(i)* perform the (SOM-like) discrete best-match search to find $\mathbf{s}_{start} = \mathbf{a}^*$ in the knot set $\mathbf{A}$, followed by *(ii)* an iterative procedure like the gradient descent. We found the Levenberg-Marquardt algorithm 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 the closest manifold point.

Fig. 3 illustrates the PSOM recall-usage in a simple $d = 3$ dimensional embedding space X, where the components $\{1, 3\}$ belong to the input space. Only these components must be specified as inputs to the PSOM. $\mathbf{w}(\mathbf{s}^*)$ is the output of the PSOM:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \square \\ x_3 \end{pmatrix} \mapsto \mathbf{s}^* \mapsto \mathbf{w}(\mathbf{s}^*) \text{ or } \begin{pmatrix} x_1 \\ w_2(\mathbf{s}^*) \\ x_3 \end{pmatrix} \qquad (4)$$

2.1 Topological Order Adds Model Bias

In the previous sections we showed the mapping manifolds for topologies which were already given. This stage of obtaining the topological correspondence includes some important aspects:

1. Choosing a topology is the first step in interpreting a given data set.

2. It introduces a strong *model bias* and reduces therefore the variance. This leads – in case of the correct choice – to an improved generalization.

3. The topological information is mediated by the basis functions. All examples shown here build on the high-dimensional extension to approximation polynomials. Therefore, the examples are special in the sense that the basis functions are varying only within their class. Other topologies can require other types of basis functions.

To illustrate this, let us consider a 2 D example with six training points. If only such a small data set is available, one may find several topological assignments. In Fig. 4 the six data points $\mathbf{w_a}$ are drawn, and two plausible, but different assignments to a 3× 2 node PSOM are displayed.

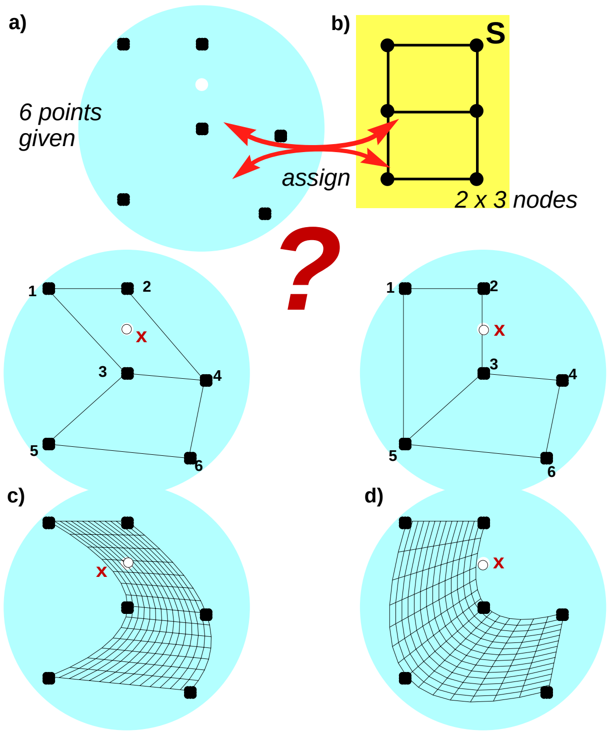


Figure 4: **The influence of topological order** shown in a data set *(a)* with ambiguous assignment to the node grid *(b)* $\mathbf{A} \in S$. Without extra knowledge, *(c)* and *(d)* are equivalently suitable. Note the test point $\mathbf{x}$ and its difference in resulting placements in the central or border region of the PSOM core mapping area. This implies differences in interpretation.

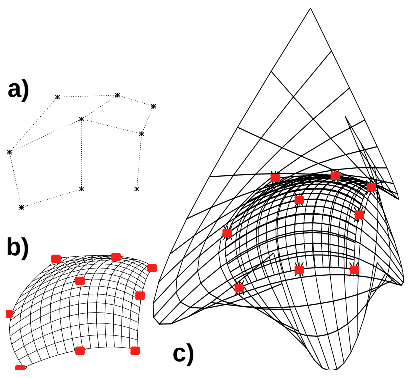


Figure 5: The mapping within the test grid *(b)* similar in size to trainings grid *(a)* is well-behaved. **Extrapolation** far outside can yield a pathological situation: *(c)* shows the test grid *(b)* plus one 1.9 times enlarged.

In the vicinity of the training data points the mapping is equivalent, but the regions interpolating and extrapolating differ, as seen for the cross-marked example query point. Obviously, it needs further information to resolve this ambiguity in topological ordering (e.g. by more samples).

2.2 Extrapolation Aspects

Now we consider the extrapolation areas, beyond the mapping region of the convex hull of the reference vectors. Fig. 5 shows a 3×3 training grid *(a)* and the X -embedding manifold *(b)* again visualized by a testgrid. *(c)* shows the superposition of a further test grid enlarged by the factor 1.9 (note, in contrast to previous illustrations with 3D-projections (Fig. 1-3) the PSOM map extrapolates here the pure 2D-problem). Here, the polynomial nature of the employed basis functions exhibits an increasingly curved embedding manifold M with growing "remoteness" to the trained mapping area. This property limits the extrapolation abilities of the PSOM, depending on the particular distribution of training data.

This leads to the advice: be suspicious, if the best-match $\mathbf{s}^*$ is found far outside the given node-set $\mathbf{A}$. A practical way is to confine the iterative search of $\mathbf{s}^*$ to a bounded region in S. In the context of the Chebyshev approximation theory advisable bounds can be stated more precisely (for details on the Chebyshev-based PSOM please consult [7, 9]).

3 Applications:

The PSOM can be favorably applied in task frames which include one or several continuous mappings (of a known topology). The training points are ordered by a SOM – or if known – can be used straight in constructing the PSOM manifold. In the latter case, the iterative "SO"-part is skipped and the "learning" is instantaneous. Thus we gain a versatile mapping tool which requires a surprisingly small number of training examples to achieve good mapping accuracy.

This section demonstrates some examples from the areas of robotics and adaptive human-machine-interfaces.

3.1 Finger Kinematics for a Robot

Fig. 6 shows one of three fingers of the hydraulically driven TUM robot hand. Its mechanical design allows roughly the mobility of the human index finger. A cardanic base joint offers sidewards gyring and full adduction with two additional coupled joints (in total 3 degree-of-freedom). For the finger control several coordinate systems are essential: the joint angles $\vec{\theta}$, the cylinder piston positions $\vec{c}$, and the relative finger tip coordinates $\vec{r}$. Further configuration dependent quantities are of interest, 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. Here we focus on the inverse kinematics – the classical hard part. Fig. 6 depicts the fingertip workspace. The "banana" form is traced out when moving the three joints on a cubical $10 \times 10 \times 10$ grid within their maximally allowed configuration.

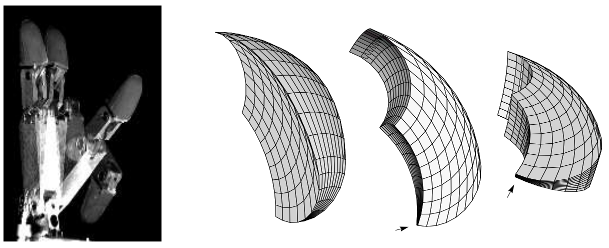


Figure 6: *(left)* stroboscopic image of one **finger** in a sequence of extreme joint positions. *(Rest)* Several perspectives of the **workspace envelope** $\vec{r}$, tracing out a cubical $10 \times 10 \times 10$ grid in the joint space $\vec{\theta}$. The arrow marks the fully adducted position, where one edge contracts to a tiny line.

We exercised several PSOMs with $n \times n \times n$ 9 dimensional data tuples $(\vec{\theta}, \vec{c}, \vec{r})$, all equidistantly sampled in $\vec{\theta}$. Fig. 7a–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. 6 should yield a rectangular grid of $\vec{\theta}$. Fig. 7c–e displays the actual result. Distortions can be visually 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.

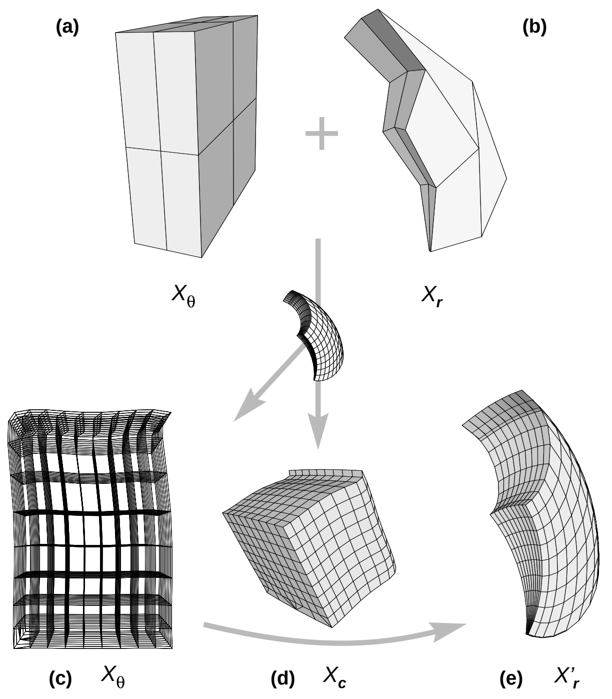


Figure 7: **Training** data set of 27 nine-dimensional points in X for the $3\times3\times3$ PSOM, shown as perspective surface projections of *(a)* the joint angle $\vec{\theta}$ and *(b)* the corresponding Cartesian sub-space (following the lines connecting the training samples allows one to verify that the "banana" really possesses a cubical topology). *(c–e)* **Inverse kinematic result** using the grid test set (=Fig. 6). *(c)* projection of the joint angle space $\vec{\theta}$ (transparent); *(d)* the stroke position space $\vec{c}$; *(e)* the Cartesian space $\vec{r}\,'$ (after forward-kinematics of θ).

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. In view of the extremely small training set displayed in Fig. 7a–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 [9, 7] for details and further comparison).

Information from Topology: 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.

The question *"why does the PSOM perform more that an order of magnitude better than the MLP?"*, leads us to the value of topological information. Fig. 8 shows the

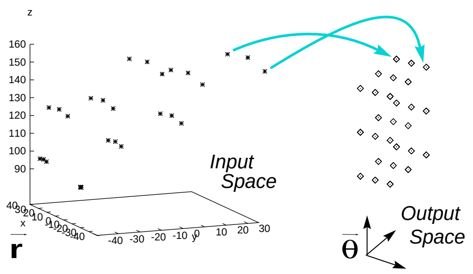


Figure 8: **A (non-PSOM) network:** The 27 training data vectors for the MLP networks: one-way from the input space $\vec{r}$ *left* to *right* the corresponding target output values $\vec{\theta}$.

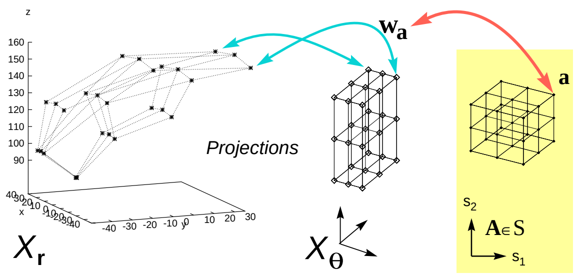


Figure 9: **PSOM case:** The same 27 training data vectors (=Fig. 8) for the bi-directional PSOM mapping: *(left)* in the Cartesian space $\vec{r}$, *(middle)* the corresponding joint angle space $\vec{\theta}$. *(Right:)* The corresponding node locations $\mathbf{a} \in \mathbf{A}$ in the parameter manifold S. Neighboring nodes are connected by lines, which reveals the "banana" structure (see Fig. 7) on the left.

27 training data pairs; on the left side, the Cartesian input space $\vec{r}$, one can recognize some zig-zag structure, but not much more. Fig. 9 depicts the PSOM situation: the PSOM gets the same data-pairs as training vectors — but additionally, it obtains the assignment to the node location **a** in the $3\times3\times3$ node grid illustrated in Fig. 9. 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 generalize to the reported positioning precision of 1%. This topological information is not available to other techniques, like the MLP or the radial basis function approach.

3.2 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 constraint functions for flexible usage.

We illustrate the flexibility of the PSOM approach in the task to position a (3 link, 4 DOF) robot

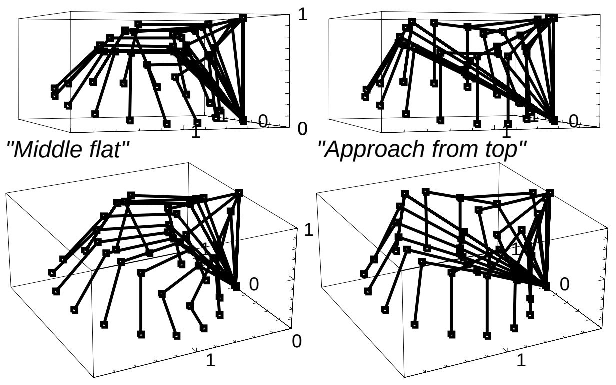


Figure 10: **Solving the Redundancy Problem:** Tracking a point in a slanted elliptical path with a 4 DOF manipulator, using a $5{\times}5{\times}5{\times}5$ $PSOM$and different extra target functions to resolve the redundancy problem. These secondary goals are reached – as far as the kinematics allows (see distal and proximal positions).

in 3D as depicted in Fig. 10. The details of this study are given in [8], here we report only the main ideas. 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 derived parameters which encode extra goal or constraint functions (here three: 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 problem in various ways. For example, the goal can be to ...
— **(Lazy & fast:)** 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;
— **(Chose angle:)** give one joint angle θ_j ($j \in \{2, 3, 4\}$) and specify $p_j > 0$;
— **(Coupled joints:)** 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;
— **(Middle flat:)** keep the middle segment horizontal: by specifying the target $x_9 = 0$ and $p_9 = 0.01$. Fig. 10*(left)* 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);
— **(Approach from top:)** grasp vertically: after specifying $x_{10} = 0$, $p_{10} = 0.01$, Fig. 10 *(right)* shows the stroboscopic tracking result.

For these different cases, there is no need for different networks, instead one single PSOM can be utilized. If we anticipate 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 their rank.

3.3 Hand Gesture Recognition by Video

The last example stays within the topic fingers and hands, but switches the application domain to human–machine–interfaces. The *GREFIT* (**G**esture **RE**cognition based on **FI**nger **T**ips) system demonstrates how a computer vision system can reliably determine the user’s hand gestures. Instead of using an expensive sensing device (like a data-glove), two standard video camera are employed to observe the user’s unadorned hand. In the first step, LLM networks detect the fingertips in one image. Second, the local stereo disparity around the five finger tips are measured (between in the two camera images, a standard procedure). And third, PSOMs are applied to map the found 2D-image information to the desired output: the 3D finger postures. Furthermore, for demonstration and user-

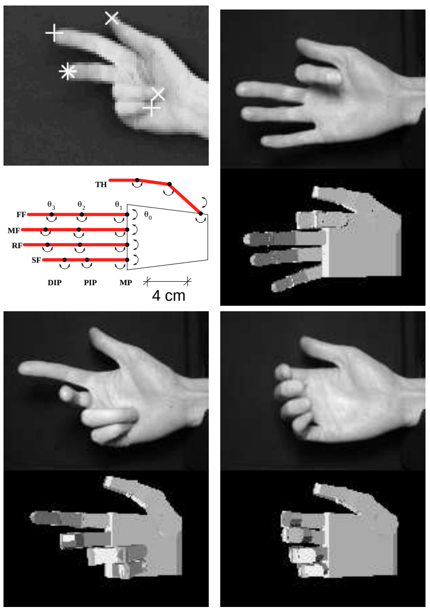


Figure 11: Hand shapes and their 3D reconstruction computed by the ***GREFIT*** system. The *top left* image depicts the fi nger tips locations by markers; *(below:)* the 16 segments and 20 revolute joints of the model hand. The other image pairs show the posture differences between the real and the rendered hand.

feedback purpose, an artificial hand model displays the posture.

Fig. 11 shows three snapshot pairs of the *GREFIT* system. The images show in every first row how one camera sees the user's hand; and in every second row the hand model with 16 segments and 20 revolving joints. The joint angles are controlled by five PSOM finger modules ($4\times 4\times 5$ PSOM). The model accounts for kinematic details and anatomical properties of the human hand (it does not include reactive deformations by external forces). By feeding hand model test data into the system, the reconstruction accuracy can be evaluated: the average error is 0.1 cm and the maximum error for all fingers is well below 0.4 cm – again a remarkable result. For more details about the 3D-reconstruction in stereo images consult [5]. As shown in [4], the system can also operate in double speed with a single camera (10 Hz on a 200 MHz PentiumPro). But the results are then less general – in the sense that it uses the presumption of a finger tendon coupling which approximates only the naturally relaxed hand.

4 Discussion and Conclusion

We presented the PSOM as versatile building 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 Kohonen's SOM learning rule, or – by construction – if the topological relation of the data is known. In numerous applications, this case can be often realized by active, structured sampling of the training data – often without any extra cost. This mechanism of incorporating prior knowledge in the network seems very attractive.

Due to the compactness of the training set, the PSOM has some overlap with fuzzy networks: 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 associative mapping concept has several attractive properties. multiple coordinate spaces can be maintained and learned simultaneously, as shown for the robot finger example. This *multi-way mapping* capability solves, e.g. the forward and inverse kinematics with the very same network. This simplifies learning and avoids worries about inconsistencies of separate learning modules. As pointed out by [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.

In the robot finger example a set of only 27 data points turns out sufficient to approximate the quite non-linear 3 D (inverse) kinematics relation with a mean positioning deviation of about 1 % of the entire workspace range.

The input selection mechanism enables to add further, parameterized target functions. These can be utilized to resolve e.g. redundancy problems which arise when the primary goal leaves a continuous solution space of possible alternatives. Here, the PSOM offers to build a battery of optimizer modules which can be learned within the same continuous associative memory. When they are activated, they can influence the best-match search in the desired manner.

In the *GREFIT* system the PSOM proved useful and accurate as well. In constrast to most other gesture recognition systems, the system is not limited to a number of static postures (e.g. sign language) but exploits the great variety of possible hand shapes using a *contiuous* parameterization of hand postures. This opens up new application areas which depend on gradual information, for instance CAD or navigating in virtual worlds.

References

[1] Mitsuo Kawato. Bi-directional neural network architecture in brain functions. In *Proc. Int. Conf. on ANN, Paris*, 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, Heidelberg, 1984.

[4] C. N¨olker and H. Ritter. GREFIT: Visual recognition of hand postures. In *Proc. Gesture Workshop* (in press), Paris, France, 1999. http://www.TechFak.Uni-Bielefeld.de/techfak/ags/ni/publicfr_99d.html.

[5] C. N¨olker and H. Ritter. Parametrized SOMs for hand posture recognition. In *Proc. IJCNN-2000* (submitted), 2000.

[6] Helge Ritter. Parametrized self-organizing maps. *Proc. Proc. Int. Conf. on ANN*, pages 568–575. Berlin, 1993.

[7] J¨org Walter. *Rapid Learning in Robotics*. Cuvillier Verlag G¨ottingen, 1996.

[8] J¨org Walter. PSOM network: Learning with few examples. In *Proc. Int. Conf. on Robotics and Automation (ICRA-98)*, pages 2054–2059, 1998.

[9] J¨org Walter and Helge Ritter. Local PSOMs and Chebyshev PSOMs – improving the parametrised self-organizing maps. In *Proc. Int. Conf. on ANN*, pages 95–102, 1995.