Neural Networks from Models to Applications
N'EURO '88, pp.579-591 L. Personnaz and
G. Dreyfus, Eds. (EZIDET,Paris, 1989)

Topology-Conserving Maps for Motor Control

Helge Ritter, Thomas Martinetz and Klaus Schulten
Department of Physics
Technical University of Munich
D-8046 Garching

Abstract:

We apply an extension of Kohonen's algorithm for the formation of topologically correct feature maps to the task of learning movements of a three-link robot arm. We find an increased robustness of the learning algorithm and an adaptive, demand driven representation of the input-output relationship as significant benefits of the topology conserving map. We suggest one-dimensional topology conserving maps as an interesting approach to trajectory formation for robots.

1.Introduction

One of the most prominent features of the brain is its organization into a collection of two-dimensional "modules" in which neighboring neurons contribute to similar tasks. These modules often represent "topology-conserving maps" in which neurons are dedicated to input data in such a fashion that the most essential interrelationships of the data are captured in the two-dimensional spatial arrangement of the corresponding neurons (for a review, see e.g. [5]). Examples are the auditive maps of sound location in the hippocampus ([3]), the motor map of eye-movements in the superior colliculus ([17]) or the crescent-shaped arrangement of motor-neuron-pools in the motor cortex innervating arm muscles ([11]).

In view of the ubiquity of these maps, a better understanding of their significance is highly desirable, not only for a better understanding of biological nervous systems but for the construction of artificial neural networks too.

A possible functional role of such maps would be to provide a matching of a two-dimensional storage and processing medium to its input and output data in such a way, that the most important communication and processing requirements can be satisfied by local interactions, spanning small distances only. In addition, there are cases where the maps are found to be capable of gradually changing the allocation of their neurons, committing neurons to more frequent stimuli at the expense of the representation of less frequent ones ([4]).

In the following, we want to present some simulation results, demonstrating the benefits of these possibilies for the solution of motor control tasks. This is a

continuation of our previous work ([12]-[16]), which is based on a model by Kohonen capable of establishing topology-conserving maps from a random sequence of sensory inputs ([6-8]). We give an extension of Kohonen's original algorithm which is capable of generating a mapping between an input and an output space and which can be applied to motor control tasks.

The resulting algorithm is applied to several tasks from robot control. In the first task, a topographic map learns the kinematics for the visuo-motor coordination of a three-link robot arm from two camera images of its end effector position. A similar approach to this problem has been taken by Kuperstein ([9],[10]). In his approach, however, the topographic organization of the maps is given beforehand and is fixed, whereas in our approach it evolves during learning. In the second task, a topographic map is used to learn the unknown dynamics of the robot arm required to execute rapid "ballistic" movements, where inertial effects and couplings between different joints become significant. The third task, hole drilling, gives some suggestion how one-dimensional topographic maps may be useful for the formation of robot trajectories.

2. The mapping algorithm

The algorithm to be described will be useful for motor tasks, which can be formulated as a continous mapping of some task specification $\mathbf{x}$ into a specification $\mathbf{y}$ of suitable motor output. We assume that both data can be represented mathematically as vectors $\mathbf{x} \in \mathbf{X}$ and $\mathbf{y} \in \mathbf{Y}$ of fixed dimensionality from suitable spaces $\mathbf{X}$ and $\mathbf{Y}$. The mapping shall be represented by an array of formal neurons, arranged as a lattice and implementing an adaptive and topographically organized look-up table. The neurons are labelled by their lattice positions $\mathbf{r}$. Two vectors, $\mathbf{w}_\mathbf{r}^{in} \in \mathbf{X}$ and $\mathbf{w}_\mathbf{r}^{out} \in \mathbf{Y}$ are associated with each neuron. The network responds to each input $\mathbf{x} \in \mathbf{X}$ by selecting the neuron $\mathbf{s}(\mathbf{x})$, for which

$$||\mathbf{w}_{\mathbf{s}(\mathbf{x})}^{in} - \mathbf{x}|| = \min_\mathbf{r} ||\mathbf{w}_\mathbf{r}^{in} - \mathbf{x}|| \tag{1}$$

and provides as output

$$\phi(\mathbf{x}) = \mathbf{w}_{\mathbf{s}(\mathbf{x})}^{out}. \tag{2}$$

We are interested in the case when there is no a-priori information about the correct input-output relationship $\mathbf{y}(\mathbf{x})$, i.e. when suitable values of the vectors $(\mathbf{w}_\mathbf{r}^{in}, \mathbf{w}_\mathbf{r}^{out})$ are not known initially. In this case we want each movement to give rise to a learning step improving the values of the vectors $\mathbf{w}_\mathbf{r}^{in}$, $\mathbf{w}_\mathbf{r}^{out}$ towards a better representation of the correct input-output relationship. In addition, we require the adjustments to be such, that neighboring neurons learn similar vector pairs $(\mathbf{w}_\mathbf{r}^{in}, \mathbf{w}_\mathbf{r}^{out})$, i.e. that

inputs and outputs get mapped in a topology preserving fashion onto the lattice. This requirement allows spatially neighboring neurons to assist each other, because they have to learn similar data. We shall see subsequently that this can significantly speed up learning and increase the robustness of the algorithm to poor initial values of the $\mathbf{w_r}$. A suitable adjustment rule for the input values is given by

$$\mathbf{w}_\mathbf{r}^{in,(new)} = \mathbf{w}_\mathbf{r}^{in,(old)} + \epsilon \cdot h_{\mathbf{rs}} \cdot (\mathbf{x} - \mathbf{w}_\mathbf{r}^{in,(old)}). \tag{3}$$

Here $\epsilon > 0$ scales the size of the learning step and $h_{\mathbf{rs}}$ is an unimodal function of the distance $||\mathbf{r} - \mathbf{s}||$ between neurons $\mathbf{r}$ and $\mathbf{s}$ which is peaked at zero argument. A convenient choice is a Gaussian

$$h_{\mathbf{rs}} = \exp(-||\mathbf{r} - \mathbf{s}||^2/2\sigma^2). \tag{4}$$

Eq.(3) can be interpreted in terms of Hebb-like synaptic modifications with active memory loss term and has been suggested by Kohonen for the formation of topology conserving feature maps of sensory signals ([6]-[8]). However, to extend this procedure to the output values $\mathbf{w}_\mathbf{r}^{out}$ we need to provide a suitable signal taking a role analogous to the sensory input $\mathbf{x}$ for the input map. To this end, we use a linear error correction rule of Widrow-Hoff type ([18]), which derives an improved estimate $\mathbf{y}^*$ of what the correct output should have been by comparing the actual outcome $\mathbf{x}^{true}$ of the movement with the desired task specification $\mathbf{x}$. Postponing the description of the error correction rule to the simulations, we then take for the output values the analogous adjustment rule

$$\mathbf{w}_\mathbf{r}^{out,(new)} = \mathbf{w}_\mathbf{r}^{out,(old)} + \epsilon' \cdot h'_{\mathbf{rs}} \cdot (\mathbf{y}^* - \mathbf{w}_\mathbf{r}^{out,(old)}). \tag{5}$$

Agiain, ϵ' scales the learning step size and $h'_{\mathbf{rs}}$ is defined similarly to $h_{\mathbf{rs}}$ before.

The maps are formed during a learning phase, consisting of a sufficiently long sequence of randomly chosen movements. During the learning phase the widths σ, σ' of h, h' and the learning step sizes ϵ, ϵ' are slowly reduced. A convenient choice for each of these parameters is

$$p(t) = p_i \cdot (p_f/p_i)^{t/t_{max}}, \tag{6}$$

where t and t_{max} are the current and maximal number of learning steps, and p_i, p_f denote initial and final values of the respective parameter. By this procedure, the system can rapidly learn the gross mapping during the early stages of learning, and include increasingly finer details towards the end (a detailed investigation of

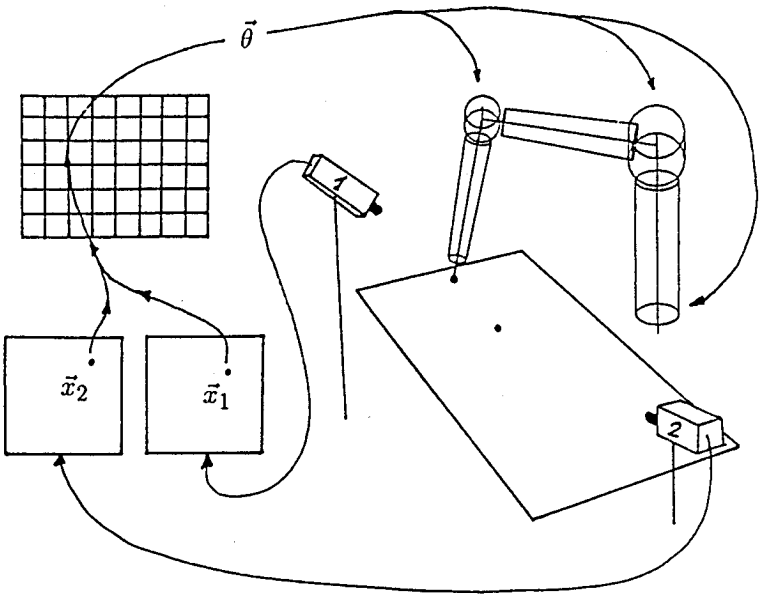


Fig.1 The simulated system. Two cameras observe the robot arm behind the table. Each camera has a quadratic image plane and maps a specified location from the scene to a pair of coordinates $\vec{x}_i$ $(i = 1, 2)$. The four-component vector $\mathbf{x} = (\vec{x}_1, \vec{x}_2)$ is fed as input to the array of neurons. The output of the array is determined by the neuron s whose vector $\mathbf{w}_s^{in}$ matches the "sensory input" $\mathbf{x}$ best.

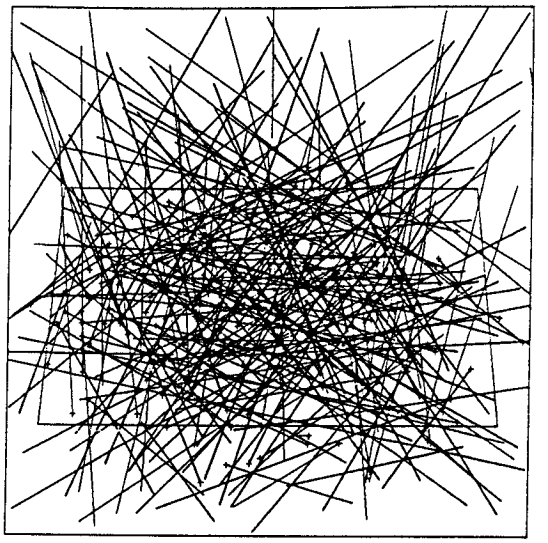
the significance of the range of $h_{\mathbf{rs}}$ and the learning step size is given in Ref.[14]). It simultaneously forms two topographic maps, one between the sampled subset of input values from $\mathbf{X}$ and the lattice, and one between the lattice and the required subset of output values from $\mathbf{Y}$. Both maps are matched in such a way as to provide an approximation $\phi(\mathbf{x})$ of the desired correct input-output relationship $\mathbf{y}(\mathbf{x})$.

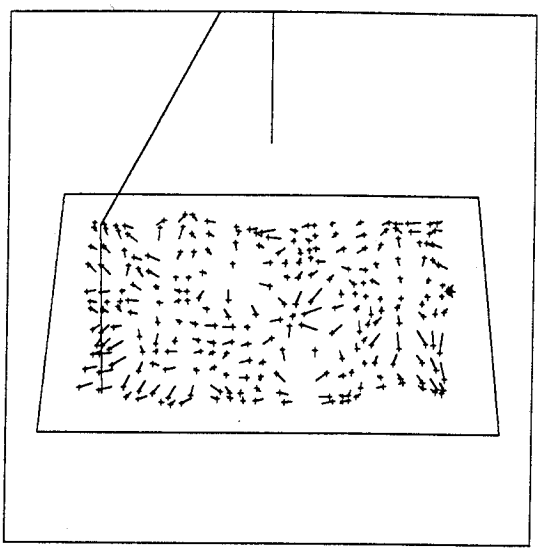
3. Simulation results

In this section we will report results from the application of the previous mapping algorithm to the control of a three-link robot arm mounted behind a table (Fig.1).

In the first simulation, the task is to learn to point to given target locations on the table. Here the task specification $\mathbf{x}$ is the desired target location. However, instead of providing cartesian coordinates, we use a pair of cameras (as shown in Fig.1) and provide the two coordinate pairs $\vec{\mathbf{x}}_1 = (x_{11}, x_{12})$, $\vec{\mathbf{x}}_2 = (x_{21}, x_{22})$ of the target location in the two camera images, i.e. we take the four-component vector $\mathbf{x} = (\vec{\mathbf{x}}_1, \vec{\mathbf{x}}_2)$ as input. The desired output of the network is a triple of joint angles $\vec{\theta} = (\theta_1, \theta_2, \theta_3)$ for which the end effector of the arm coincides with the target location. No prior information about the geometry of the arm (apart from having three joint angles), the placement of the cameras and their imaging characteristics is provided to the network.

Following section 2, we associate with each neuron a four-component vector $\mathbf{w}_\mathbf{r}^{in}$, whose components specify two image points, one in each camera image. To improve the accuracy of the representation, we use each output vector $\mathbf{w}_\mathbf{r}^{out}$ not only to





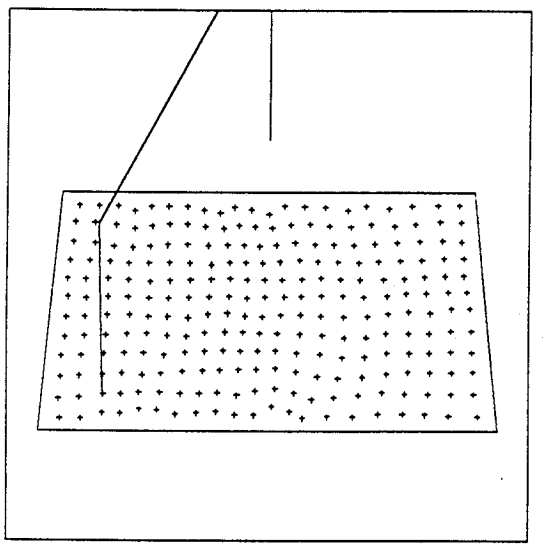


Fig.2a-c Error of end effector positions as seen from camera 1. For each of the 12 x 21 target positions associated with $\mathbf{w}_{\mathbf{r}}^{in}$ the resulting end effector location is indicated by a cross mark. The mismatch to the intended target position is indicated by an appended line segment. From top left to bottom left: (a) Large initial errors, reflecting random initialization of the system. (b) After 4000 trial movement only small errors remain. Furthermore, target positions outside the table surface, being never required during training, are no longer represented by neurons. (c) After 20000 iterations, no visible deviations between target and achieved end effector locations remain.

represent three joint angles $\vec{\theta}_{\mathbf{r}}$, but also a 3×4 Jacobian $\mathbf{A}_{\mathbf{r}}$. Hence $\mathbf{w}_{\mathbf{r}}^{out}$ is the pair $(\vec{\theta}_{\mathbf{r}}, \mathbf{A}_{\mathbf{r}})$, arranged as a 15-dimensional vector. The Jacobian $\mathbf{A}_{\mathbf{r}}$ shall transform the residual deviation $\mathbf{x} - \mathbf{w}_{\mathbf{s}(\mathbf{x})}^{in}$ between the sensory input $\mathbf{x}$ and the best matching vector $\mathbf{w}_{\mathbf{s}(\mathbf{x})}^{in}$ of the input map into a linear correction of the output angles $\vec{\theta}_{\mathbf{s}(\mathbf{x})}$ associated to $\mathbf{w}_{\mathbf{s}(\mathbf{x})}^{in}$. Therefore, the response of the network to a sensory input $\mathbf{x}$ from the cameras will be given by

$$\vec{\theta} = \vec{\theta}_{\mathbf{s}(\mathbf{x})} + \mathbf{A}_{\mathbf{s}(\mathbf{x})}(\mathbf{x} - \mathbf{w}_{\mathbf{s}(\mathbf{x})}^{in}). \tag{7}$$

To derive from each trial movement improved estimates $\vec{\theta}^*$ and $\mathbf{A}^*$ for joint angles

and Jacobian, each movement is separated into two phases: (i) A "gross movement phase", where the contribution of the Jacobian is neglected and the joint angles are set to $\vec{\theta}_{\mathbf{s}(\mathbf{x})}$. The corresponding pair of locations of the end effector in the two camera images is denoted by the four component vector $\mathbf{x}_I$. (ii) A "fine movement phase", correcting the joint angles to their final values given by (7). The resulting image locations of the end effector are denoted by $\mathbf{x}_F$. Then

$$\vec{\theta}^* = \vec{\theta}_{\mathbf{s}} + \mathbf{A}_{\mathbf{s}}(\mathbf{x} - \mathbf{x}_F), \tag{8}$$

$$\mathbf{A}^* = \mathbf{A}_{\mathbf{s}} + \mathbf{A}_{\mathbf{s}}(\mathbf{x} - \mathbf{w}_{\mathbf{s}} - \mathbf{x}_F + \mathbf{x}_I)(\mathbf{x}_F - \mathbf{x}_I)^T ||\mathbf{x}_F - \mathbf{x}_I||^{-2} \tag{9}$$

can be shown to provide an improved estimate $\mathbf{y}^* = (\vec{\theta}^*, \mathbf{A}^*)$ for the adaptation step (5) of the output map ([16]).

The results of a simulation with a lattice of 12×21 neurons are shown in Figs.2a-c. Any error in the input-output relationship represented by the network manifests itself in a mismatch between the desired target location and the end effector position achieved by the network. Fig.2a shows these mismatches from the view of camera 1 for the initial network, which was initialized with random numbers. The resulting mismatches are shown for the subset of target locations which belong to the discretization points $\mathbf{w}_{\mathbf{r}}^{in}$ of the neurons. Each mismatch is depicted as a line segment joining the desired target location with the actually achieved end effector location (cross mark). The fairly long line segments indicate a very unsatisfactory initial performance of the network. However, after 4 000 movements to randomly chosen target points on the table, only small mismatches remain (Fig.2b). In addition, the discretization points chosen by the network have retracted to the table surface exclusively, i.e. only the region to which movements are actually requested is represented. Finally (20 000 movements, Fig.2c) target positions and end effector positions agree to within the resolution of the diagrams. For a real system of mean size of 1m the positioning error now would have fallen below a value of 1mm.

In the previous simulation, the topology conserving map has learnt to associate arm postures to visually designated target locations. This is a valid approach to movement control as long as inertial effects, i.e. the dynamics of the movement, can be neglected. Such effects, however, come into play for rapid movements and lead to couplings between different joints (see e.g. [1]). The next simulation addresses this problem for the case of so-called "ballistic" movements, which are initiated by a brief torque pulse and proceed freely thereafter. The task to be learnt by the network is to specify a torque pulse $\mathbf{d}(t) = (d_1(t), d_2(t), d_3(t))$ (d_i =torque for joint i) which accelerates the end effector to a desired velocity $\mathbf{v}$. Taking $\mathbf{d}(t)$ to be a delta-function in time,

$$\mathbf{d}(t) = \vec{\tau} \cdot \delta(t), \tag{10}$$

we need to specify its "vectorial amplitude" $\vec{\tau} = (\tau_1, \tau_2, \tau_3)$, which will depend on $\mathbf{v}$ and on the current arm configuration $\vec{\theta}$. Therefore, a straightforward approach would be to take the 6-component vector $(\mathbf{v}, \vec{\theta})$ as input $\mathbf{x}$, and $\vec{\tau}$ to be the desired output of the network. However, as a general feature of Newton's equations of motion, there is a linear relationship

$$\vec{\tau} = \mathbf{A}(\vec{\theta})\mathbf{v} \tag{11}$$

between $\vec{\tau}$ and the desired velocity $\mathbf{v}$, involving a configuration dependent matrix $\mathbf{A}(\vec{\theta})$. This admits the following much more convenient representation. Input is $\mathbf{x} = \vec{\theta}$ and output is $\mathbf{A}(\vec{\theta})$, from which $\vec{\tau}$ can be easily obtained via (11). Hence, each vector $\mathbf{w}_\mathbf{r}^{in}$ specifies three joint angles, and each vector $\mathbf{w}_\mathbf{r}^{out}$ contains nine elements specifying a 3×3 matrix $\mathbf{A_r}$. Whenever neuron $\mathbf{s(x)}$ was selected for a movement, an improved estimate $\mathbf{A}^*$ of $\mathbf{A_{s(x)}}$ is given by

$$\mathbf{A}^* \;=\; \mathbf{A_s} \;+\; \frac{1}{||\bar{\mathbf{v}}||^2}(\vec{\tau} - \mathbf{A_s}\bar{\mathbf{v}})\bar{\mathbf{v}}^T. \tag{12}$$

Here $\vec{\tau} = \mathbf{A_s}\mathbf{v}$ is the torque used to accelerate the end effector to the desired velocity $\mathbf{v}$, and $\bar{\mathbf{v}}$ is the velocity actually obtained.

Figs.3a-c show the results of a simulation with 360 neurons, arranged as a 15×24-lattice, controlling the (simulated) robot arm of Fig.1. As mass distribution we took three point masses of equal magnitude, located at the end effector and the two outer joints of the arm (this gives stronger inertial couplings than for the case of a more uniform mass distribution). The initial values $\mathbf{w}_\mathbf{r}^{in}$ were chosen to correspond to random locations of the end effector in the table surface, and for the outputs $\mathbf{w}_\mathbf{r}^{out}$ we chose the correct matrices plus a random error of 25% (relative to the euclidean norm $||\mathbf{w}_\mathbf{r}^{out}||$) per matrix element. As a direct visualization of the 9-dimensional output map is not possible, we show instead the reaction of the end effector to test movements. Each figure is a perpendicular view of the table surface and the arrows indicate the velocities of the end effector obtained from 360 pairs of test movements, each pair originating from one of the end effector locations associated with the vectors $\mathbf{w}_\mathbf{r}^{in}$. Each test movement required to accelerate the end effector from rest to unit velocity, directed parallel to one of the table edges. Fig.3a shows the poor outcome of the initial test movements, resulting from the random initialization of the system. After 1 000 trial movements (acceleration of end effector to randomly directed unit velocity from randomly chosen location of table surface) the performance is already much better (Fig.3b), until after 10 000 trial movements the accuracy of the test movements is very good (Fig.3c).

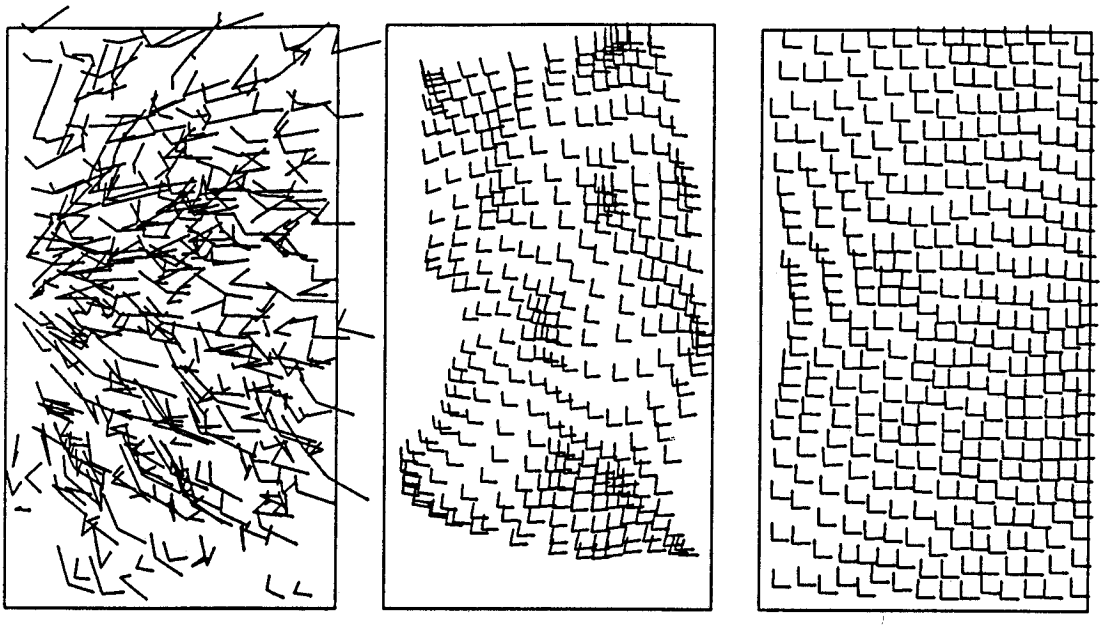


Fig.3a-c Learning of ballistic movements. Each diagram shows the resulting velocities (indicated by arrows) of the end effector to a set of test movements. For each of the end effector locations associated with the 360 arm postures specified by $\mathbf{w}_{\mathbf{r}}^{in}$ acceleration of the end effector from rest to unit velocity along the two table edges was required. From left to right: (a) initial poor performance, reflecting initialization with random numbers. (b) after 1 000 trial movements reactions begin to resemble task. (c) after 10 000 trials test movements are executed accurately.

For both simulations, participation of neighboring neurons in each adjustment step, effected by the nonzero width of the Gaussians $h_{\mathbf{rs}}$ and $h'_{\mathbf{rs}}$, was found to be an essential ingredient for good convergence. For the input map, this ensures that neighboring neurons become "responsible" for similar task specifications $\mathbf{x}$. Consequently they have to learn similar outputs by the assumed continuity of the input-output mapping. Hence participation of a whole subpopulation of neighboring neurons in each adaptation step is a rudimentary form of generalization, speeding up learning and made possible by the topographic organization. In addition, it greatly increases robustness to poor initial values, as false adjustments from different neurons with poor outputs tend to cancel, whereas the spatially slowly varying correct parts from different neurons accumulate. As an example, running the second simulation from the same initial data, but restricting each neuron to adaptation steps resulting from its own output only, yields (again after 10 000 iterations) the result shown in Fig.4. Only part of the neurons has managed to learn the correct responses and the performance compares very poorly with Fig.3c. This behaviour is shown in a more quantitative fashion in Fig.5, which shows the average velocity error $||\mathbf{v} - \bar{\mathbf{v}}||$ versus the number of trial movements for three different initial widths σ'_i (in lattice units) of the Gaussian $h'_{\mathbf{rs}}$ (the final width was $\sigma'_f = 0.2$ for all three

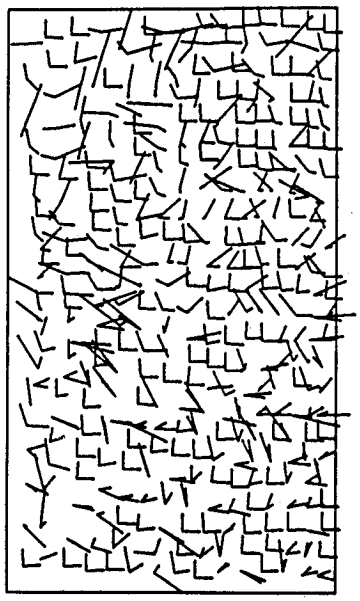


Fig.4 Resulting poor convergency in the absence of cooperation between the neurons. Shown is the end result of a repetition of the simulation of Fig.3. The only difference was restriction of each adjustment step to the selected neuron only (i.e. $h'_{rs} = \delta_{rs}$). As a consequence, only very few neurons managed to learn the correct matrix $\mathbf{A_r}$.

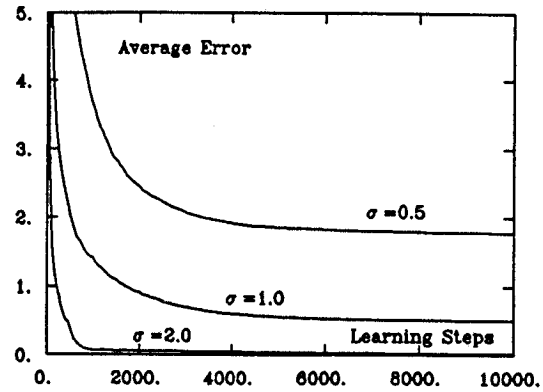


Fig.5 Influence of cooperation between neurons on convergence. The diagram shows the average velocity error of the achieved end effector velocity versus the number of movement trials for three different initial values of σ_i. For weak cooperation (i.e. small values of σ_i) a significant residual error remains.

cases). For the two cases with small initial neighborhoods, the significant residual error indicates that only a fraction of the neurons manages to achieve convergence to correct output values. Only the simulation with $\sigma'_i = 2$ shows good convergence. A mathematical analysis of this behavior is possible (Ref.[13]) and shows, that the inclusion of neighboring neurons in each learning step results in an effective increase of the set of initial values from which convergence to the correct output matrices occurs.

A further important problem in robot control is the formation of trajectories. A trajectory can be considered as a one-dimensional topology-conserving map between a line and the phase space of the robot arm. Learning a good trajectory for a repetitive task can therefore be formulated as formation of a topology-conserving map. An immediate application is provided by the solution of the "Travelling Salesman Problem", based on a topology conserving map as first suggested by Durbin and Willshaw ([2]). In terms of a robotics application, this approach can be used to yield close to optimal task space trajectories for the "hole drilling problem": A robot is required to repetitively drill a set of holes with fixed, prespecified positions

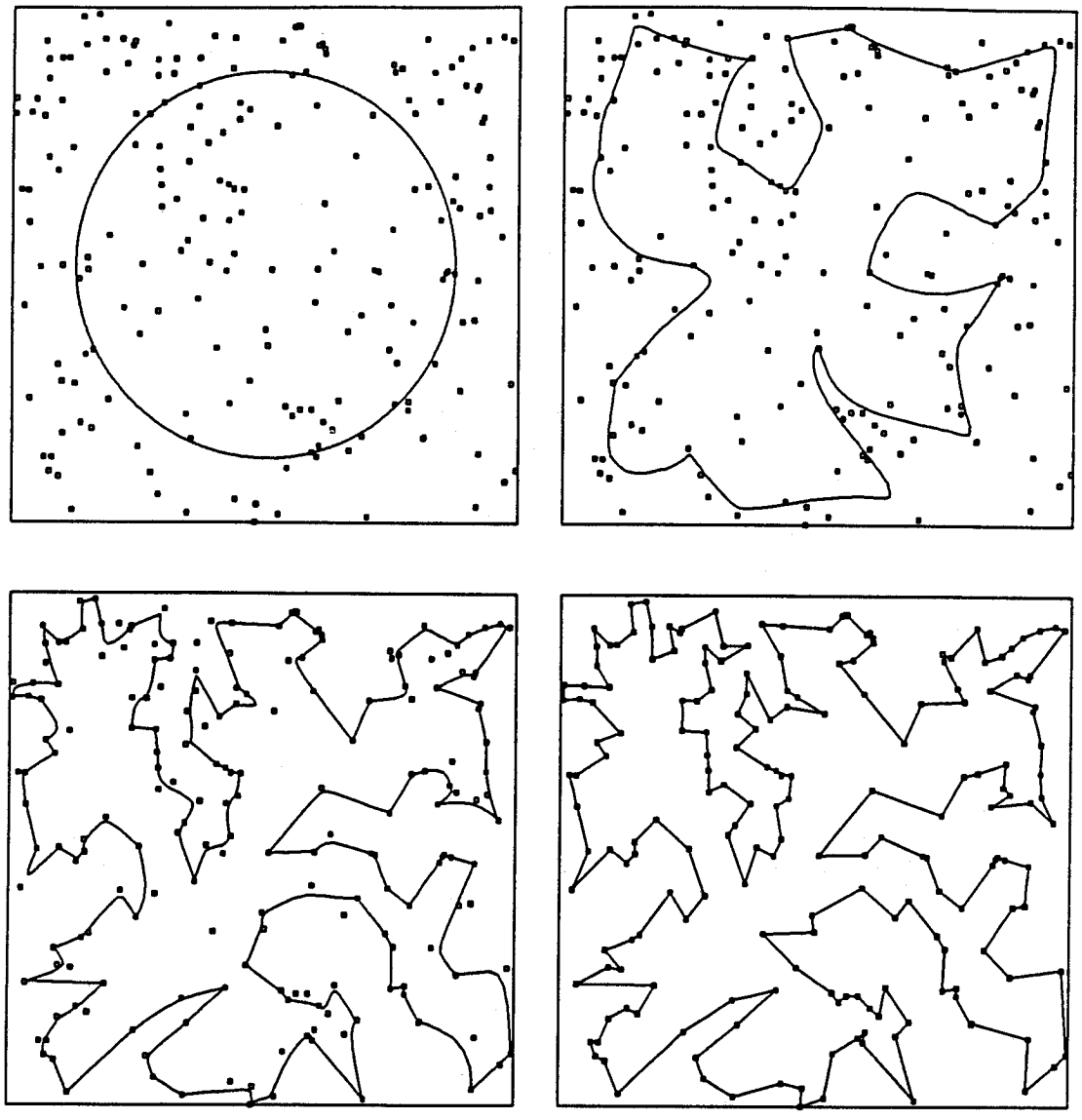


Fig.6 Gradual formation of trajectory for visiting a set of 200 randomly positioned holes on a square workpiece. Initial trajectory is assumed circular (top left) and comes close only to a small subset of all holes. Intermediate trajectories (after 5 000 and 7 000 adaptation steps) do increasingly better. Final trajectory (after 10 000 adaptation steps) visits all holes and has a close-to-optimal length of 10.98

on a workpiece. A good trajectory will minimize the total distance travelled by the arm to drill all holes. Such trajectory can be learnt by visiting the holes in random order for some time and taking each visited hole position $\mathbf{x}$ as input to a set of "neurons" arranged as a closed chain instead of a lattice. Each adaptation step is performed according to (5) (vectors $\mathbf{w}_r^{out}$ are not needed in this case) and the linear

sequence of positions $\mathbf{w}_{\mathbf{r}}^{in}$ as $\mathbf{r}$ passes around the chain specifies a discretized version of the current trajectory learnt so far.

Fig.6 shows the development of a trajectory for a square workpiece with 200 randomly positioned holes. The final trajectory has a length of 10.98 (assuming a unit square). Although it is not the global minimum (using an annealing method, we found trajectories which are a few percent shorter), it is reasonably close to an optimal path. Modifying (4) to optimize measures other than euclidean distance, e.g. curvature or energy dissipation, is an interesting direction for further research.

4.Conclusion

Based on an algorithm of Kohonen for the self-organized formation of topologically correct feature maps, we have developed an extension of the original approach which can be applied to motor learning tasks. Necessary prerequisite for the application of the algorithm is a continuous relationship between input and output signals. Random movement trials are used to gradually build two topology-conserving maps on the same neural sheet simultaneously. One map provides an image of the sensory inputs specifying the desired movements. The other map is a representation of the required motor outputs. Both maps are matched such that each input signal gets associated to the correct motor output. We present results from two simulations of the algorithm for the control of a three-link robot arm. In the first simulation, the task is to learn the arm postures associated with visually designated target locations for the end effector. The second task requires learning suitable arm torque pulses for accelerating the end effector to a specified velocity. The topographic organization of the input-output map for both applications allows generalization between spatially neighboring neurons. In particular, neighboring neurons can participate in their learning steps, which greatly improves speed and robustness of convergence to the correct input-output mapping. As a further promising field for algorithms based on topographic maps, we suggest the problem of trajectory formation for robots. Finding a trajectory can be viewed as the formation of a one-dimensional topographic map. As an illustrative problem, which is feasible with the present algorithm, we present the hole drilling problem.

Acknowledgement: This work has been supported by the German Ministry for Science and Technology (ITR-8800-G9).

References:

[1] Brady M., Hollerbach J.M., Johnson T.L., Lozano-Perez T., Mason M.T. (1984): Robot Motion: Planning and Control, MIT Press, Cambridge, Massachusets.

[2] Durbin R, Willshaw D (1987) An analogue approach to the travelling salesman problem using an elastic net method. Nature 326:689-691

[3] Harris W.A. (1986) Learned topography: the eye instructs the ear. TINS, March 1986, 97-99

[4] Jenkins WM, Merzenich MM, Ochs MT (1984) Behaviorally controlled differential use of restricted hand surfaces induces changes in the cortical representation of the hand in area 3b of adult owl monkeys. Society for Neurosciences Abstracts, 10, 665.

[5] Knudsen E.I., du Lac S., Esterly S.D. (1987) Computational maps in the brain. Annual Reviews of Neuroscience 10:41-65

[6] Kohonen T. (1982a) Self-organized Formation of Topologically Correct Feature Maps. Biological Cybernetics 43:59-69

[7] Kohonen T. (1982b) Analysis of a Simple Self-organizing Process. Biological Cybernetics 44:135-140

[8] Kohonen T. (1982c) Clustering, Taxonomy and Topological Maps of Patterns. Proceedings of the 6th Int. Conf. on Pattern Recognition, Munich pp 114-128

[9] Kuperstein M. (1987) Adaptive Visual-Motor Coordination in Multijoint Robots using Parallel Architecture. Proc. IEEE Int. Conf. Automat. Robotics 1595-1602, Raleigh NC

[10] Kuperstein M. (1988) Neural Model of Adaptive Hand-Eye Coordination for Single Postures. Science 239:1308-1311

[11] Murphy J.T., Kwan H.C., MacKay W.A., Wong Y.C. (1977) Spatial Organization of Precentral Cortex in Awake Primates. III. Input-Output Coupling. Journal of Neurophysiology 41:1132-1139

[12] Ritter H., Schulten K. (1986) Topology Conserving Mappings for Learning Motor Tasks. In Denker J.S. (Ed), Neural Networks for Computing, AIP Conference Proceedings 151, Snowbird, Utah, pp. 376-380

[13] Ritter H., Schulten K. (1987) Extending Kohonen's Self-Organizing Mapping Algorithm to Learn Ballistic Movements. In Eckmiller R. and von der Malsburg C. (Eds.), Neural Computers, Springer, Heidelberg, pp. 393-406

[14] Ritter H., Schulten K. (1988a) Convergency Properties of Kohonen's Topology Conserving Maps: Fluctuations, Stability and Dimension Selection. Biological Cybernetics, in press.

[15] Ritter H., Schulten K. (1988b) Kohonen's Self-Organizing Maps: Exploring their Computational Capabilities. IEEE ICNN 88 Conference, San Diego

[16] Ritter H., Martinetz T., Schulten K. (1988c) Topology Conserving Maps for Learning Visuo-motor-Coordination. Submitted to Neural Networks.

[17] Robinson D.A. (1972) Eye Movements Evoked by Collicular Stimulation in the Alert Monkey. Vision Research 12 1795-1808

[18] Widrow B., Hoff M.E. (1960) Adaptive switching circuits, WESCON Conv Record, part IV pp. 96-104