Hierarchical Neural Net for Learning Control of a Robot's Arm and Gripper

Thomas M. MARTINETZ and Klaus J. SCHULTEN

Beckman Institute and Department of Physics
University of Illinois at Urbana-Champaign
405 North Mathews Ave., Urbana, IL 61801, USA

Abstract: We introduce a hierarchical neural network structure capable of learning the control of a robot's arm and gripper. Based on Kohonen's algorithm for the formation of topologically correct feature maps and on an extension of the algorithm for learning of output signals, a simulated robot arm system learns the task of grasping a cylinder. The network architecture is that of a 3-dimensional cubic lattice in which is nested at each lattice node a 2-dimensional square lattice. The robot learns without supervision to position its arm and to properly orient its gripper by observing its own trial movements. In our simulation, the error in positioning the manipulator after training was 0.3% of the robot's dimension, and the residual error in orienting the gripper was 3.8°. Due to cooperation between neighboring neurons during the training phase, less than two trial movements per neuron were sufficient to learn the required control tasks.

Keywords: Hierarchical neural net, robotics, gripper, motor control, Kohonen, topology conserving maps.

1. Introduction

In many motor tasks a hierarchical order of basic movements can be recognized. For example, in the comparatively simple task of grasping a cylinder by a robot arm, two hierarchically ordered movements are required. First, the gripper has to be positioned in the vicinity of the object by a proper posture of the whole arm. Second, the gripper needs to be oriented relative to the object such that the object can be grasped. This subsequent movement depends not only on the object's orientation in space, but also on the posture of the arm to which the gripper is attached. In contrast to other biologically inspired neural network models [1], which solve the task of grasping an elongated object by treating both of these movements in a homogenously structured network, we employ a neural network structure which reflects the intrinsic hierarchy of the two underlying movements.

The simplest approach would be to control the positioning movement and the subsequent orientation of the gripper independently. This approach is sufficient only as long as the object is presented in a workspace which is small compared to the dimensions of the robot arm, for instance, if the object is always located in a small region of the visual field. In this case the different arm postures, which correspond to different target locations, do not vary over a broad range and can be approximated as being constant, with the result, that the required movements to orient the gripper depend only on the object's orientation. In this case, two separate neural network controllers, one for the positioning movement and one for the orientation of the manipulator, are sufficient.

This is no longer valid for the situation we are considering. In Fig.1 we show the robot arm with the gripper and two cameras which provide the visual information. The workspace, which is indicated schematically in Fig.1, has an extent which requires a broad range of arm postures in order to reach all possible object locations. To account for the dependence of gripper orientations on arm postures we employ a hierarchical neural network structure which consists of a main net for the control of the arm posture, and several subnets, which learn to coordinate the subsequently performed orientation of the gripper at different areas of the workspace.

In Section 2, we describe the adaptation scheme used by the main neural net to learn the arm positioning movement, and we introduce an adjustment rule for the joint angles of the robot arm, based on a feedback

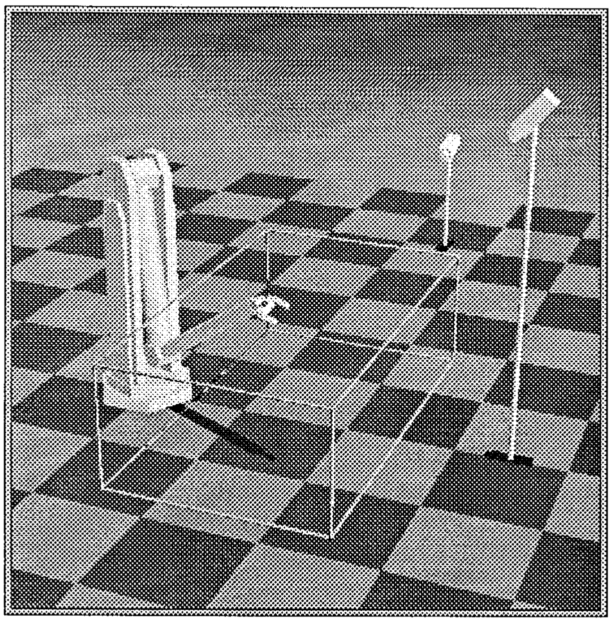

Fig.1: Model of the simulated robot. The arm has three degrees of freedom. The gripper is able to move within the vertical plane and around its own axis. The two cameras provide the visual information necessary for the neural network controller to perform the required movement and to adapt to the robot's characteristics.

loop, which is able to reduce the residual positioning error to an arbitrary small value. In Section 3, the subnets for the gripper orientation and their hierarchical arrangement within the main net is described. In this section, we also explain the hierarchically organized cooperation between neurons, which leads to the very high training efficiency which requires less than two learning steps per neuron. In the last section we present the results of our simulation.

2. Neural Net for Arm Posture

For each trial movement of the robot we present a cylinder at a randomly chosen location within the workspace and with a random orientation of its axis. The neural network controller then has to process two kinds of input signals, one for the positioning subtask, i.e. for adopting a proper arm posture, and one for the orientation of the gripper. In Fig.2, we depict schematically the images of the cylinder seen by the two cameras. The image coordinates of the center of the bars as seen through camera 1 and camera 2 are denoted by u_{x1}, u_{y1} and u_{x2}, u_{y2}, respectively, and the size of each bar along the x- and y-axis is denoted by x_{x1}, x_{y1} and x_{x2}, x_{y2}. As decribed in more detail in our previous work [2,3], u_{x1}, u_{y1} and u_{x2}, u_{y2} can be combined to form a four-dimensional vector $\mathbf{u}$ which carries the information about the spatial location of the cylinder. In a similar way, x_{x1}, x_{y1} and x_{x2}, x_{y2} can be used by the neural network to infer the cylinder's orientation.

In this section we explain briefly the coordination and the learning of the positioning movement. The cylinder is presented at randomly chosen locations within the workspace. All the resulting input signals $\mathbf{u}$ serve to attempt an arm posture which positions the gripper in front of the object to be grasped. The inputs $\mathbf{u}$ form a three-dimensional submanifold within the four-dimensional input space, and, therefore, we use a neural net of a three-dimensional topology to represent this submanifold. By employing Kohonen's algorithm for the formation of topologically correct feature maps [4], a topographic mapping between the relevant part of the input space and the three-dimensional neural net develops during learning. This mapping can be seen as a discretization of the three-dimensional submanifold of input signals, attributing to each node of the 3-D Kohonen net a position $\mathbf{w_s}$. The transformation $\vec{\theta}(\mathbf{u})$ between the target location $\mathbf{u}$ and the required three joint angles $\vec{\theta}$ of the arm can be linearized in the vicinity of each discretization point and be presented by a vector $\vec{\theta}_\mathbf{s}$ together with a tensor $\mathbf{A_s}$.

To each discretization point $\mathbf{w_s}$ corresponds a formal neuron $\mathbf{s}$ of the Kohonen-network; the neuron associates a location $\mathbf{w_s}$ within the four-dimensional input space with a three-dimensional vector $\vec{\theta}_\mathbf{s}$ of joint

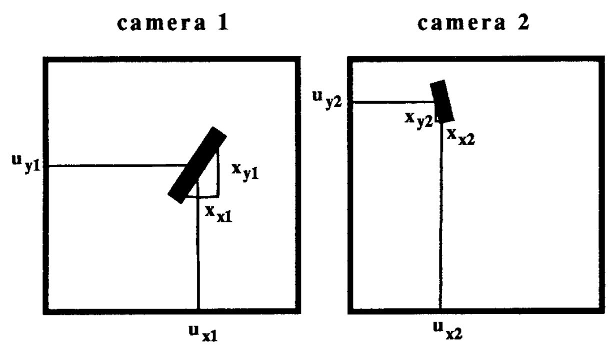

Fig.2: Schematic view of the cylinder in camera 1 and camera 2. The center of each bar, the projection of the cylinder onto the focal plane of camera 1 and camera 2, is denoted by its coordinates u_{x1}, u_{y1} and u_{x2}, u_{y2}, respectively. The information about the orientation of the cylinder is provided by the the size of each bar in the x and y direction of each focal plane, i.e. by x_{x1}, x_{y1} and x_{x2}, x_{y2}.

angles and a 3×4 matrix $\mathbf{A_s}$. Angles $\vec{\theta}_\mathbf{s}$ and matrices $\mathbf{A_s}$ are used to position the end effector in front of the object. The discretization occurs in a topologically ordered manner, meaning that neighboring neurons of the Kohonen-network are associated with neighboring subsets of the input space [2,3].

After a cylinder has been presented to the robot and the center location $\mathbf{u}$ abstracted, it is determined which neuron is currently responsible for the arm positioning. This neuron s is identified by the condition that the input $\mathbf{u}$ is closer to the discretization point $\mathbf{w_s}$ than to any other $\mathbf{w_r}$, $\mathbf{r} \neq \mathbf{s}$. The positioning of the end effector consists then of two phases. In a primary gross positioning phase the robot joints assume the angles which are given by the components of the three-dimensional vector $\vec{\theta}_\mathbf{s}$. The position of the end effector in the "retinas" of both cameras after this gross movement will be denoted by a four-dimensional vector $\mathbf{v}_i$, which is already close to the four-dimensional retinal target location $\mathbf{u}$. In a secondary correction step we now use the linear correction

$$\Delta\vec{\theta} = \mathbf{A_s}(\mathbf{u} - \mathbf{v}_i) \tag{1}$$

for a corrective joint movement (see also [5]). Here $\mathbf{A_s}$ is the Jacobian of the transformation $\vec{\theta}(\mathbf{u})$ at the discretization point s, and $\Delta\vec{\theta}$ is usually sufficient to correct the error of the gross movement very precisely [5]. The final position of the gripper is seen by the cameras at a location $\mathbf{v}_f$. In departing from the procedure adopted in [2,3] we iterate the correction several times, in case the error is still too large for the desired purpose. This is done by taking the new difference $\mathbf{u} - \mathbf{v}_f$ between the last movement and the target, and correcting the joint angles again by using the Jacobian $\mathbf{A_s}$. One obtains $\Delta\vec{\theta} = \mathbf{A_s}(\mathbf{u} - \mathbf{v}_f)$ for the second correction and so forth. In our simulation, described in more detail in Section 4, this correction was carried out three times per trial movement.

Each trial is accompanied by an improvement of $\vec{\theta}_\mathbf{s}$ and $\mathbf{A_s}$. For this purpose we use a linear error correction rule of the Widrow-Hoff-type [6], leading to improved estimates [5]

$$\mathbf{A}^* = \mathbf{A_s} + ||\Delta\mathbf{v}||^{-2} \cdot \mathbf{A_s}(\mathbf{u} - \mathbf{v}_f)\Delta\mathbf{v}^T, \tag{2}$$

$$\vec{\theta}^* = \vec{\theta}_\mathbf{s} + \mathbf{A_s}(\mathbf{w_s} - \mathbf{v}_i). \tag{3}$$

Because of the topologically correct assignment of the neurons to the inputs, adjacent elements of the three-dimensional grid have to adjust to similar output values, and, therefore, $\vec{\theta}^*$ and $\mathbf{A}^*$ are used to improve the output values in a whole neighborhood of s. This neighborhood is defined by a function $h_\mathbf{rs}$ which is unity at $\mathbf{r} = \mathbf{s}$ and decays to zero as $\mathbf{r}$ deviates from s. The resulting adjustments are

$$\mathbf{A}_\mathbf{r}^{new} = \mathbf{A}_\mathbf{r}^{old} + \epsilon h_\mathbf{rs}(\mathbf{A}^* - \mathbf{A}_\mathbf{r}^{old}), \tag{4}$$

$$\vec{\theta}_\mathbf{r}^{new} = \vec{\theta}_\mathbf{r}^{old} + \epsilon h_\mathbf{rs}(\vec{\theta}^* - \vec{\theta}_\mathbf{r}^{old}). \tag{5}$$

This cooperation between neighboring neurons during the training phase provides an enormous increase in speed and stability of the convergence of the learning algorithm [2].

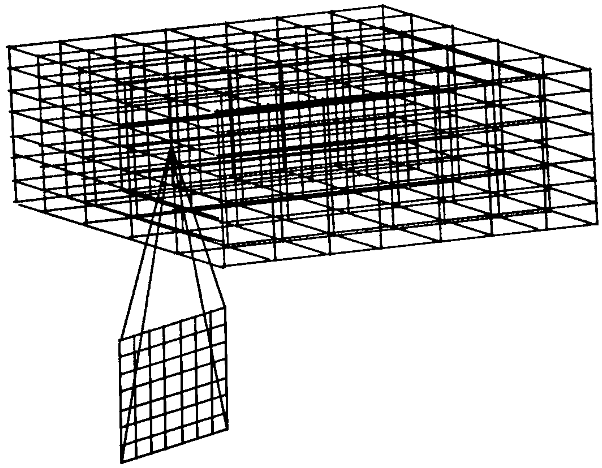

Fig.3: The hierarchical neural net architecture. To each element of the three-dimensional arm posture control net a two-dimensional subnet is assigned. If s is the element of the three-dimensional net selected to control the positioning of the arm, the assigned subnet S_s gets activated to coordinate the orientation of the gripper.

3. Neural Nets for Gripper Orientation

To be able to extract the orientation of the cylinder in order to orient the gripper, the neural network controller uses $\bar{\mathbf{x}} = norm(x_{x1}, x_{y1}, x_{x2}, x_{y2})$ as its input signal. Since one needs only information on the orientation of the cylinder and not on its length, the four-dimensional vector $(x_{x1}, x_{y1}, x_{x2}, x_{y2})$ needs to be known only in regard to its orientation.

The gripper orientation is defined by two angles denoted by the 2-dimensional vector $\vec{\phi}$. For the same orientation of the cylinder but different arm postures, the neural network controller has to produce different pairs of angles for the orientation of the gripper. Because of this "coupling" between arm posture and gripper orientation, two seperate networks for each movement are not sufficient in case high accuracy is required. A possible network structure which accounts for this "coupling" is shown in Fig.3. To each element s of the three-dimensional (arm posture control) neural net we assign a small subnet S_s for controlling the orientation of the gripper within the small area of the workspace, for which element s is responsible. To learn the transformation $\vec{\phi}_s(\bar{\mathbf{x}})$ from input signals $\bar{\mathbf{x}}$ to the required angles $\vec{\phi}$ of the manipulator, each subnet S_s uses the same learning scheme already employed for the positioning movements. Again, Kohonen's vector quantization algorithm [4] forms a topologically correct mapping from the input space X of input signals $\bar{\mathbf{x}}$ to the subnet S_s, and adaptation steps of the same form as used for the positioning movement are employed to learn the required output values.

For the discretization of the space X of input signals $\bar{\mathbf{x}}$ we choose nets of a two-dimensional topology because the orientation of the cylinder has two degrees of freedom, and, therefore, the relevant submanifold of actual input signals $\bar{\mathbf{x}}$ is two-dimensional. For the control of the two angles of the gripper, we assign to each element $\mathbf{q}$ of subnet S_s a two-dimensional vector $\vec{\phi}_{\mathbf{qs}}$ for the gross orientation of the gripper and a tensor $\mathbf{B}_{\mathbf{qs}}$ of dimension 2×4 for subsequent corrective fine movements. After the selection of element s, vector $\vec{\phi}_{\mathbf{qs}}$ and tensor $\mathbf{B}_{\mathbf{qs}}$ of the discretization point $\mathbf{q}$ which is closest to the input $\bar{\mathbf{x}}$ are chosen to perform the orientation of the gripper.

For learning the output values $\vec{\phi}_{\mathbf{qs}}$ and $\mathbf{B}_{\mathbf{qs}}$, adaptation steps of the same form already used for $\vec{\theta}_s$ and $\mathbf{A}_s$ are employed. First, by a steepest descent learning rule of the Widrow-Hoff type (see Eq. (2) and (3)) improved estimates $\vec{\phi}^*$ and $\mathbf{B}^*$ are determined. Since the elements of subnet S_s are again assigned to the input space X in a topologically correct manner, the result can be used to adapt the output values in a whole neighborhood of neuron $\mathbf{q}$ (see eq. (4) and (5)). Additionally, because the subnets are also arranged in a topologically correct way within the three-dimensional grid, the learning success of neuron $\mathbf{q}$ may also be "spread" to neighboring subnets. Adjacent subnets have to learn similar transformations $\vec{\phi}_s(\bar{\mathbf{x}})$, and, therefore, may benefit from sharing adjustments of $\vec{\phi}_{\mathbf{qs}}$ and $\mathbf{B}_{\mathbf{qs}}$. This hierarchical cooperation between neurons within a subnet and between subnets in the main net can be expressed mathematically by

$$\mathbf{B}_{\mathbf{rp}}^{new} = \mathbf{B}_{\mathbf{rp}}^{old} + \epsilon h_{\mathbf{rs}} \delta g_{\mathbf{pq}} (\mathbf{B}^* - \mathbf{B}_{\mathbf{rp}}^{old}), \tag{6}$$

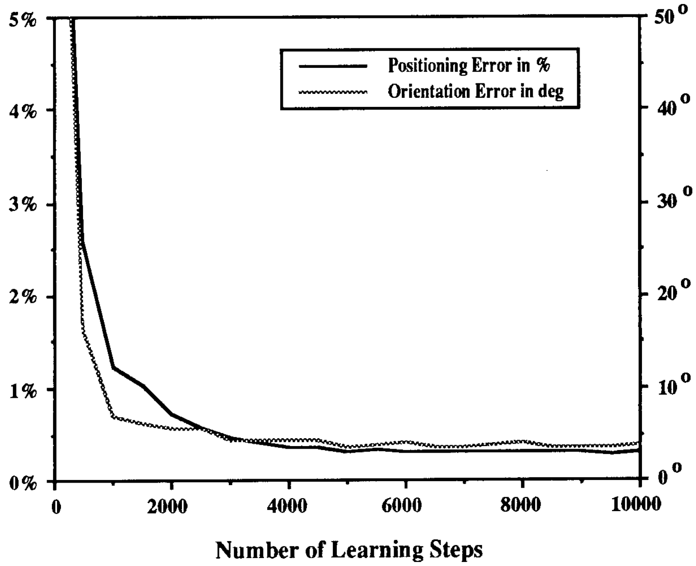

Fig.5: Positioning error and orientation error as a funktion of the number of learning steps. After 5000 trial movements, the errors are already close to their final values. These final values, reached after 10 000 adaptation steps, are 0.3% and 3.8°, respectively.

$$\vec{\phi}_{\mathbf{rp}}^{new} = \vec{\phi}_{\mathbf{rp}}^{old} + \epsilon h_{\mathbf{rs}} \delta g_{\mathbf{pq}} (\vec{\phi}^* - \vec{\phi}_{\mathbf{rp}}^{old}). \tag{7}$$

The tensor $h_{\mathbf{rs}}$ defines the range of cooperation between subnets and was already used to determine the neighborhood in the adaptation steps (4) and (5). The second tensor $g_{\mathbf{pq}}$ describes the neighborhood between units within each subnet and is of the same form as $h_{\mathbf{rs}}$, namely unity at $\mathbf{p} = \mathbf{q}$ and vanishing as the distance between $\mathbf{p}$ and $\mathbf{q}$ increases. The factors ϵ and δ scale the size of the adaptation steps and decrease as learning proceeds [2,3].

This hierarchical cooperation between the neurons and between the subnets leads to a significant acceleration of the learning process. Since each neuron within the employed network structure has many neighbors with which it can share adjustment steps, it is sufficient for the neural net to perform less than two learning steps per neuron to obtain the simulation results described in the next section. Although the number of neurons for learning the control of the orientation of the gripper in addition to arm positioning is 25 times as high as the number of neurons for arm positioning alone [3,5], the total number of required trial movements does not increase when the gripper capability is added to the arm.

4. Simulation Results

In this last section we describe the results of our simulation. We used a three-dimensional lattice of 7 x 12 x 4 elements for arm positioning and subnets of 5 x 5 elements for gripper orientation. The initial output values $\vec{\theta}_{\mathbf{s}}$, $\mathbf{A_s}$, $\vec{\phi}_{\mathbf{sq}}$, $\mathbf{B_{sq}}$ as well as the initial discretizations $\mathbf{w_s}$ and $\mathbf{w_{qs}}$ were chosen at random.

Figure 5 shows the performance errors for arm positioning and for gripper orientation as a function of the number of performed trial movements. After only 1000 learning steps the positioning error has already decreased below 1.2% of the dimensions of the workspace, and the deviation of the orientation of the gripper relative to the axis of the cylinder is already lower than 6.8°. After 10 000 trial movements, the positioning error and the orientation error have decreased to 0.3% and 3.8°, respectively. As we can recognize in Fig.5, both errors are very close to their final values after only 5 000 learning steps. The rapid decay of both errors is mainly due to the corrective fine movements, which are performed three times after each gross positioning. The Jacobians $\mathbf{A_s}$ and $\mathbf{B_{sq}}$ are adapting to values close to their final ones already after several hundred learning steps. This leads to fine movements which are able to correct possible bad gross positionings already at the beginning of the learning procedure.

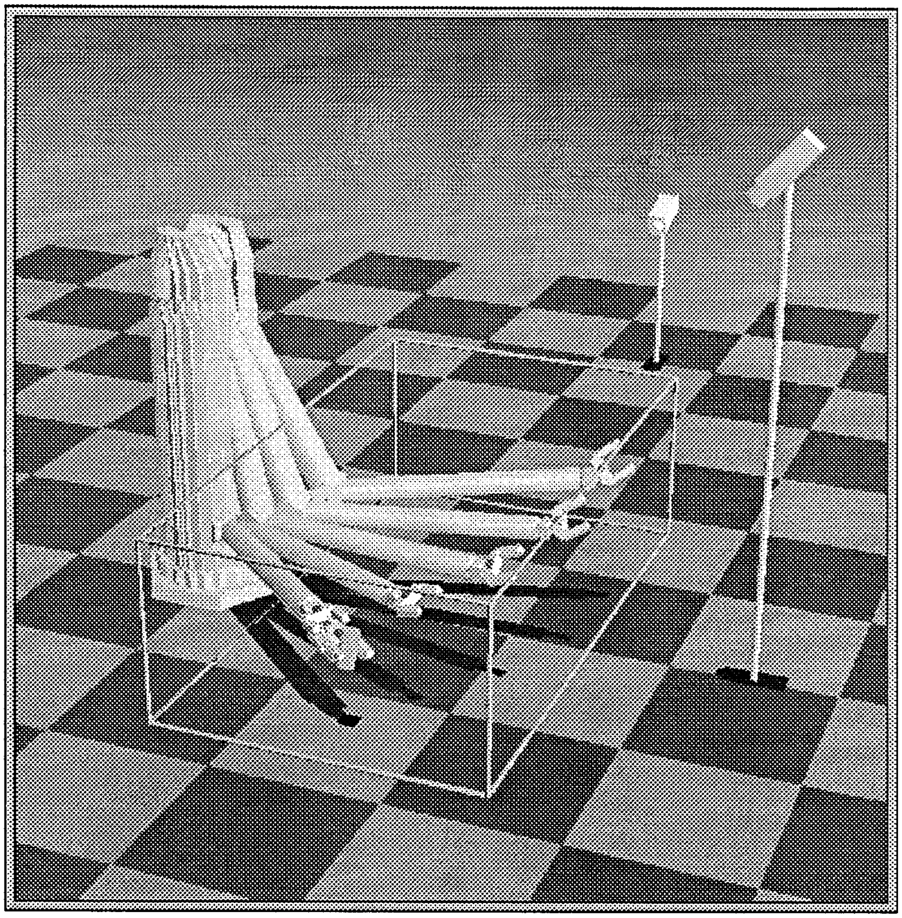

Fig.6: Successive arm postures of a successful trial to grasp the cylinder.

To demonstrate the final result, we show in Fig.6 successive frames of one successful trial to grasp the cylinder. It is possible to discern the gross arm positioning and the successive fine movements. Simultanously, the orientation of the gripper is adjusted, which enables the robot to grasp the object.

Acknowledgement

The authors would like to thank Helge Ritter for many fruitful discussions. This work has been supported by the University of Illinois and by a fellowship of the Volkswagen Foundation to T.M..

References

[1] Kuperstein M., Rubinstein J. (1989) Implementation of an Adaptive Neural Controller for Sensory-Motor Coordination. IEEE Control Systems Magazine, Vol.9, No.3, pp. 25-30.

[2] Ritter H., Martinetz T., Schulten K. (1988) Topology-Conserving Maps for Learning Visuomotor-Coordination. Neural Networks 2, pp. 159-168.

[3] Martinetz T., Ritter H., Schulten K. (1989) 3D-Neural Net for Learning Visuomotor-Coordination of a Robot arm. IJCNN-89, Conference Proceedings, Washington 1989, Vol.II:351-356.

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

[5] Martinetz T., Ritter H., Schulten K. (1990) Learning of Visuomotor Coordination of a Robot Arm with Redundant Degrees of Freedom. ICNC-90, Conference Proceedings, Düsseldorf 1990, (in press)

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