# Topology Representing Maps and Brain Function

Klaus Schulten and Michael Zeller
*Theoretical Biophysics Group*
*Beckman Institute for Advanced Science and Technology*
*University of Illinois at Urbana-Champaign*
*Urbana-Champaign, IL 61801, USA*

[Nova Acta Leopoldina NF 72, Nr. 294, 133–157 (1996)]

## INTRODUCTION

The brain, the most complex organ of higher organisms, has attracted the attention of theoretical scientists for as long as it was recognized as the seat of the intellect. A recent renaissance of the theory of the brain has been prompted by experimental advances on the one side and by the advent of the modern computer on the other side, the latter serving as an engine to numerically simulate scenarios suggested by various theories.

The modern computer, however, turned out to be more than a number cruncher, it became a metaphor for the brain and it spawned a new discipline, the science of information. Even though this science is a youngster among the established disciplines, given its young age one cannot but be impressed with its achievements. One such achievement, considered below, builds on the discovery of a close relationship between algorithms and geometry and led to the field of computational geometry.

The brain is a computer, a fact which is trivial and, hence, nearly useless. We understand the brain's hardware only on a very rough scale, we definitely do not understand it's software, we are seeing glimpses of how it codes information. Nevertheless, that the brain computes is a fact and one may gain some inside from a comparison with its engineered brethren, the computer. For this purpose we ask what computational strategies information science suggests for solutions to problems with which the brain is confronted. Surely, one has to exercise great caution in translating answers into statements regarding neural structures and neural processes, but not pursuing such questions and answers is tantamount to turning a blind eye to available knowledge.

In this lecture we look towards information science and ask two questions. First, how can the brain use its main structural and dynamic components, neurons and their synapses, to code the extremely wide ranging information it is confronted with. We will turn towards computational geometry for an answer and demonstrate that the theory of Delaunay tesselations provides a natural framework in which brain maps can be described. This approach is then compared to the representation of visual input in area v1 of the visual cortex as observed and modelled.

We also pose a second question regarding the brain's overarching objective to provide a rational link between sensory input and motor action, e.g., between sight and flight. This extremely ambitious question requires a justification which we derive from the fallacies of lesser goals, which slice out of the overall function of the brain partial capabilities. Such reduction raises serious questions: Does the chosen capability really constitute a significant step for the overall function, i.e., of generating an optimal response to sensory inputs, or has it been chosen just because a solution is at hand? Does one understand how the capability studied and the solution suggested could

link to other necessary capabilities in the chain of brain processes which realize the overarching goal. The questions raised are avoided if one models a brain function initiated by sensory data and completed by an appropoiate motor action.
Naturally we focus on a most simple task which we choose as the task of grasping a cylindrical object through visual guidance. In lieu of a proper animal model we attempt to solve this task for an engineered camera – robot system, choosing a robot arm which shares properties with a skeletal muscle system. Our approach which combines visual input and motor responses is formulated. The camera–robot system is described and the application of the theory to this system is demonstrated.

## TOPOLOGY REPRESENTING NETWORKS

In this section we consider the problem prevelant for brain function, how the brain's neurons and their synapses can represent data structures pertaining, e.g., to an extremely wide variety of sensory modalities. We begin with the observation that even the extremely large number of neurons in the brains of higher animals, about $10^9$, and their synapses, about $10^{12}$, are no match to the combinatorical multitude of data which need to be processed in an animal's lifetime. Definitely, the brain must account for the continuum of data through a discrete and very sparse representation. We assume the data to be embedded in a Euclidean space $\Re^D$ of dimension $D$. The data inherit from this space, in particular, a metric which serves to compare data. The problem of representing continuous data $\mathbf{v}$ through a discrete set $S = \{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ of representative data is commonly referred to as vector quantization: one partitions the relevant volume of the Euclidean space, where data points $\mathbf{v}$ occur with significant frequency, into a discrete and finite set of cells, chosen as polyhedra; each point in the space is then represented by the center $\mathbf{w}_j$ of the cell $V_j$, the so-called Voronoi polyhedron in which the data point lies. The set of Voronoi polyhedra $V_j$ which partitions the space are defined through the set $S$ as follows

$$V_i = \{\mathbf{v} \in \Re^D \,\big|\, \|\mathbf{v} - \mathbf{w}_i\| \leq \|\mathbf{v} - \mathbf{w}_j\| \; j = 1, \ldots, N\} \qquad i = 1, ..., N \; . \tag{1}$$

The Voronoi polyhedra provide a complete partitioning of the embedding space $\Re^D$, i.e., $\Re^D = \bigcup_{i=1}^{N} V_i$. The set of all Voronoi polyhedra is called the Voronoi diagram. An example arises in case of a triangulation of a plane in which case the Vorono polyhedra are triangles. This is illustrated in Fig. 1, bottom, where the grey lines represent a section of the Voronoi diagram.
Figure 1 illustrates how the neurons of the brain on the one side and a data structure as it occurs, for example, for a sensory modality on the other side, are matched through the Voronoi diagram. The centers of the Voronoi polyhedra can be naturally identified with the neurons of the brain, the interior of the polyhedra with the receptive fields of the neurons. A sensory event corresponds to the presentation of a data point $\mathbf{v} \in \Re^D$ to the algorithm which seeks to develop the Voronoi diagram. One determines the Voronoi polyhedron$V_j$ in which $\mathbf{v}$ lies, i.e., the $\mathbf{w}_j$ closest to $\mathbf{v}$, and interpretes this as implying the excitation of neuron $j$. It must be stressed from the outset that the generality of this approach is misleading: nature certainly does not consider complex sensory impressions, e.g., a complete visual scene like a face, as a data point; rather it filters from such impressions elementary facets which are re-combined in the brain to reach interpretations of, e.g., visual scenes. The wisdom of biological evolution manifests itself in the type of filters applied; in

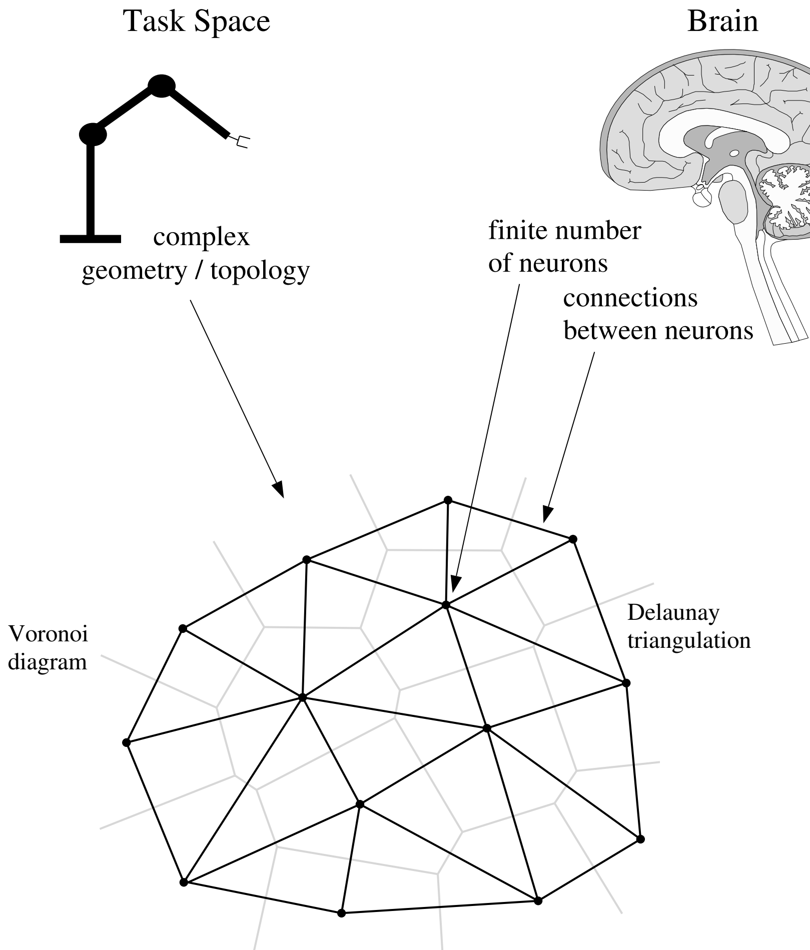


FIG. 1. How does the brain represent a complex task space while beeing limited by a finite number of neurons and connections between them? Using vector quantization, it is possible to map continous data on a discrete set of cells. Each point in the space is then represented by a Voronoi polyhedron. Further explanations are given in the text.

rare cases, for example in vision, the filters are known and the approach suggested can be applied in a straightforward way. In other cases, suitable filters must be chosen first, e.g., through a principal component analysis applied to local data features (RUBNER and SCHULTEN 1990, RUBNER et al. 1990), before the present approach can be applied.

Nevertheless, the receptive field structure provides a powerful solution to a basic geometrical problem which deals with the proximity of points in a metric space as it arises in many information processing tasks. The most prominent example of such a *proximity problem* is the *nearest-neighbor* or, more generally, the *$k$-nearest-neighbor search*: given $N$ points in a metric space, which is (are) the nearest ($k$ nearest) neighbor(s) to a new given query point (DUDA and HART 1973). This *best match retrieval* has to be performed in classification and interpolation tasks with applications in areas ranging from speech- and image processing over robotics to efficient storage and transfer of data (MAKHOUL et al. 1985, KOHONEN et al. 1984, NAYLOR and LI 1988, GRAY 1984, NASRABADI and KING 1988, NASRABADI and FENG 1988, RITTER and SCHULTEN 1986).

For many data processing purposes it is necessary to develop also a representation of neighborhood relationships of the data structure. The simplest task which requires such representation is

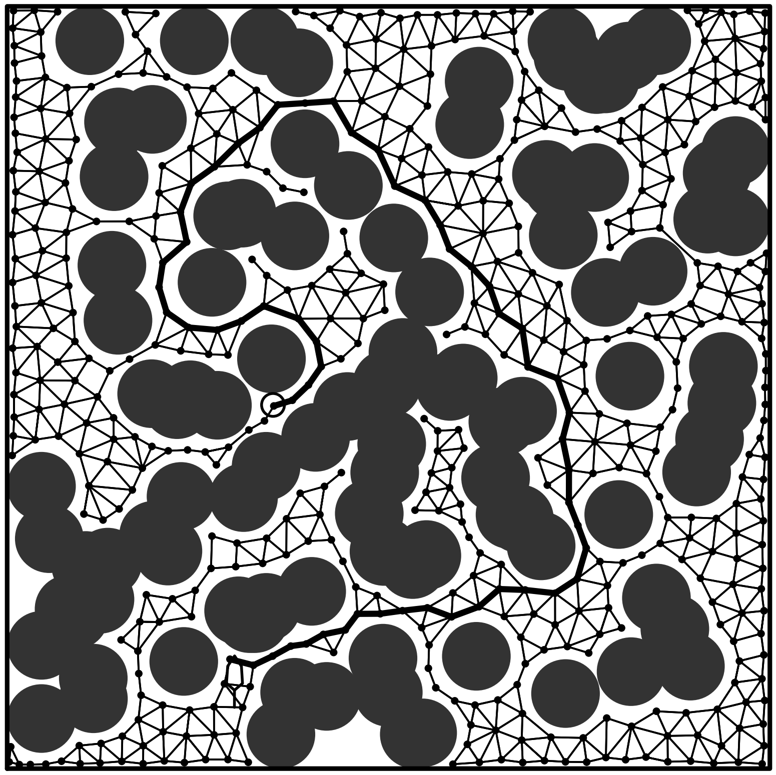

FIG. 2. A "computer mouse" (lower left corner) and the topology preserving map of the environment it has formed by employing the competitive Hebb rule. The dark areas are obstacles the "computer mouse" had to circumvent while exploring its environment by random walk. Succesive positions of the "computer mouse" formed the input patterns for the network. The distribution of the pointer positions $\mathbf{w}_i$, which are marked as dots, is dense. Hence, the connectivity structure formed by the competitive Hebb rule corresponds to the masked Delaunay triangulation and defines a topology preserving map of the feature manifold, i.e., the obstacle-free area. The topology preserving map represents the topology of the obstacle-free part such that the map can be used by the "computer mouse" to plan short paths to target locations (e.g., the location marked by the circle). (MARTINETZ and SCHULTEN 1994).

that of finding the shortest path between two data points of a continuous data structure. If one wants to determine such path within the framework of Voronoi diagrams one needs to envoke the so-called Delaunay tesselation which connects the centers of all Voronoi polyhedra with the centers of neighboring Voronoi polyhedra. The latter are defined as those polyhedra $V_i$ and $V_j$ which share a vertex, edge, face, etc, such that the property $V_i \cap V_j \neq \emptyset$ holds. The ensuing Delaunay tesselation is also illustrated in Figure 1. This representation allows one to associate the minimum path between two data points in $\Re^D$ with the shortest path in the Delaunay tesselation. This task is rendered non-trivial due to the possiblity that the data structure embedded in $\Re^D$ often does not fill a convex volume, but rather a volume rendered strongly corrugated through obstacles. An example of such situation is presented in Fig. 2 which presents within the Delaunay triangulation a minimum path betweeen two points. The Delaunay triangulation has its neurobiological counterpart in the synapses between nerve cells as we discuss in the next Section.

In a plane, the Delaunay tesselation is actually a triangulation and is obtained if one connects all pairs $\mathbf{w}_i, \mathbf{w}_j \in S$, the Voronoi polygons $V_i, V_j$ of which share an edge. In general, for embedding spaces $\Re^D$ of arbitrary dimension $D$, the Delaunay triangulation $\mathcal{D}_S$ of a set $S = \{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$

of points $\mathbf{w}_i \in \Re^D$ is defined by the graph whose vertices are the $\mathbf{w}_i$ and whose *adjacency matrix* $\mathbf{A}$, $A_{ij} \in \{0,1\}, i,j = 1,...N$ carries the value one if and only if $V_i \cap V_j \neq \emptyset$. Two vertices $\mathbf{w}_i, \mathbf{w}_j$ are connected by an edge if and only if their Voronoi polyhedra $V_i, V_j$ are adjacent.

A number of theorems about properties of the Voronoi diagram and the Delaunay triangulation are known (see, e.g., PREPARATA and SHAMOS 1985). However, most of them are valid or at least can be proven only in the planar case, for $D = 2$. In higher dimensional embedding spaces $\Re^D, D > 2$ only little is known so far. One reason is that only for $D = 2$ the Voronoi diagram and the Delaunay triangulation are planar graphs and, therefore, only for $D = 2$ Euler's formula can be applied (BOLLOBÁS 1979). Euler's formula provides the important result that in the planar case the number of edges of the Voronoi diagram as well as of the Delaunay triangulation does not exceed $3N - 6$ and, hence, the Voronoi diagram and the Delaunay triangulation can be stored in only linear space (linear in the number of vertices $N$). Further, due to this result, both structures are transformable into each other in only linear time.

A generalization of the minimum path problem is posed by the construction of the *Euclidean minimum spanning tree*: given $N$ points in a metric space, what is the graph of minimum total length whose vertices are the given points (KRUSKAL 1956, PRIM 1957, DIJKSTRA 1959). Constructing the Euclidean minimum spanning tree is a common task in applications requiring optimally designed networks, e.g., communication systems which have minimal interconnection length. Other applications of the Euclidean minimum spanning tree are in clustering (GOWER and ROSS 1969, ZAHN 1971), pattern recognition (OSTEEN and LIN 1974), and in searching for (approximate) solutions of the *traveling salesman problem* (ROSENKRANTZ et al. 1974).

Voronoi diagrams and Delaunay tesselation arise also in the *triangulation problem*: given $N$ points in a plane, connect them by non-intersecting straight lines so that every region inside the convex hull of the $N$ points is a triangle. The triangulation problem occurs in the finite-element method (STRANG and FIX 1973) and in function interpolation on the basis of $N$ data points where the function surface is approximated by a network of triangular facets (GEORGE 1971). A comprehensive overview of the above and further proximity problems can be found in PREPARATA and SHAMOS (1985). Delaunay triangulations have recently been applied in computational fluid dynamics (BRAUN and SAMBRIDGE 1995).

Constructing the Delaunay triangulation in a preprocessing stage yields a starting point for efficiently solving proximity problems. It can be shown that if the Delaunay triangulation of a given set of points $S$ is known, the above stated and other proximity problems can be solved with at most linearly increasing computational effort. The triangulation problem, for example, is obviously already solved with the construction of the Delaunay triangulation and does not need further computation[1]. The computation time needed for finding the Euclidean minimum spanning tree is reduced significantly since the edges of the Euclidean minimum spanning tree are a subset of the edges of the Delaunay triangulation (SHAMOS 1978). Knowing the Delaunay triangulation, it only requires $\mathcal{O}(N)$ instead of $\mathcal{O}(N \log N)$ time for its construction. The nearest-neighbor and $k$-nearest-neighbor search can be performed in only $\mathcal{O}(\log N)$ instead of $\mathcal{O}(N)$ time by exploiting

---

[1]Solving the triangulation problem by means of the Delaunay triangulation has advantages particularly in function interpolation. When a function is approximated piecewise-linear over the facets of a triangulation, the Delaunay triangulation yields a smaller worst case error than any other triangulation (OMOHUNDRO 1990). This property will be exploited for the motor control problem below.

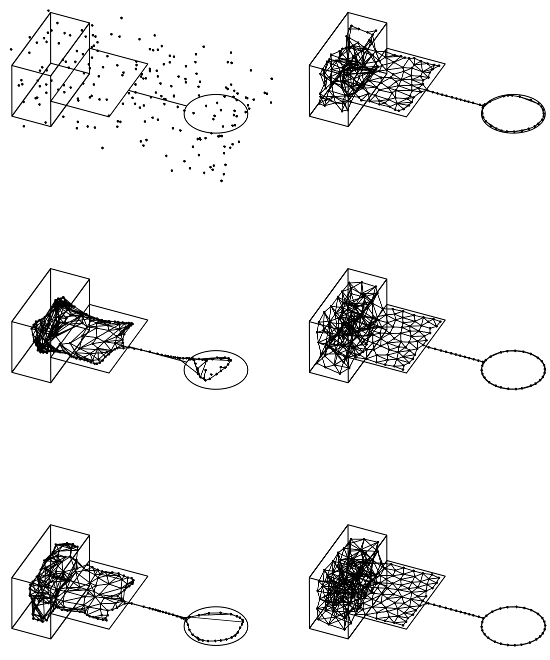

FIG. 3. The competitive Hebb rule together with the neural gas algorithm forming a topology preserving map of a topologically heterogeniously structured manifold. The given manifold **M** consists of a three-dimensional (right parallelepiped), a two-dimensional (rectangle), and a one-dimensional (circle and connecting line) subset. The *neural gas* algorithm as an efficient input driven vector quantization procedure distributes the pointers over the manifold **M**. Depicted are the initial state, the network after 5000, 10000, 15000, 25000 and 40000 adaption steps. At the end of the adaption procedure the n etwork (graph) forms a perfectly topology preserving map that reflects the topological structure and the dimensionality of the manifold **M**. (MARTINETZ and SCHULTEN 1994).

the Delaunay triangulation (KNUTH 1973).

We have recently developed an algorithm which achieves the construction of a Delaunay tesselation for data sets with a metric, but unknown dimension of the embedding space. Data points are sequentially fed to the algorithm which develops a Voronoi diagram and Delaunay tesselation which continue to adapt to the data set (MARTINETZ and SCHULTEN 1994). An example has been shown in Fig. 2 above. Another example is presented in Fig. 3. The figure shows the evolution of the Delaunay tesselation of a data structure embedded in $\Re^3$, composed of a cube, a plane, a line and a circle. A sequence of data points $\mathbf{v}(t),\ t = 1, 2, \ldots$ are randomly generated in the mentioned object; initially, "neurons" are floating disconnected in the vicinity of this object, then enter it, spread across it and finally fill it and tesselate it, the cube by a "foam" of tetraeders, the plane by a "carpet" of triangles, and line and circle by a respective one-dimensional graph.

**DEVELOPMENT OF MAPS IN THE VISUAL CORTEX**

In the preceeding section we discussed how a set of neurons, labelled by $j = 1, 2, \dots N$, and their synapses, collected in a synaptic matrix $C_{ij}$, $i, j = 1, 2, \dots N$, provide through a Voronoi diagram and Delaunay tesselation a discrete representation of a data structure and its neighborhood properties. We want to demonstrate now that this construction has a counterpart in the cortical areas of the brains of higher organisms.

The most studied part of the cortex is the visual cortex, and therein area v1. This area, which exists on both sides of the brain, is connected through nerve fibres to the lateral geniculate nuclei (one for each side) and from there to the retinas of the eyes (LEE and MALPELI 1994, TZONEV et al. 1995). Let us denote, for the time being, the relevant neurons in area v1 by $j = 1, 2, \dots N$. Applying naively the considerations in the previous section one would expect that a neuron, say neuron $j$, will be activated when a visual impression is perceived, i.e., when a cat sees a mouse sitting in the laboratory. This not so; many neurons become activated by such impression.

The question arises which data are then represented by the activity of single neurons. The answer is that the cortical neurons actually represent local visual stimuli. HUBEL and WIESEL (1962) have shown that the cortical neurons represent (i) the location of a visual stimulus in the left or right eye described, say, by a number $e$, (ii) the location of the stimulus in the visual field as projected on the retinas, represented by coordinates $x, y$, (iii) an directional anisotropy of the stimulus represented by a variable $\phi$ with periodicity of 180°. This latter feature, *a priori* is unexpected, but arises naturally when one carries out a principal component analysis of natural images which are composed to a large degree of line segments. How such analysis can be realized by neural networks is demonstrated in (RUBNER and SCHULTEN 1990, RUBNER et al. 1990).

We will see that the brain needs to pay a price for filtering directional anisotropy into its primary visual representation and, apparently, does so to achieve already some preprocessing in the primary visual representation. Since neurons in v1 are not necessarily responding only to stimuli of one eye, the variable $e$ above is not Boolean, but may be chosen from a finite interval of the real numbers, say $[-1, 1]$, where a value of -1 (1) corresponds to a stimulus associated solely with the left (right) eye.

Applying the earlier considerations we conclude at this point that cortical neurons in area v1 represent data embedded in a space resulting from the Cartesian product of a three-dimensional torus $T$ and the interval $[-1, 1]$. For the sake of simplicity we will neglect the latter factor in the following discussion, i.e., consider a hypothetical cortical area connected solely to one eye; our actual comparision with biological data,however, will include the complete data space.

At this point we take notice of the fact that cortical neurons are actually characterized through a specific location in the cortex which can be considered for this purpose as a sheet with a coordinization $\vec{\rho} = (\xi, \zeta)^T$. This implies that the Voronoi diagram establishes a map which assigns to positions $\vec{\rho}_j$, $j = 1, 2, \dots N$ centers $\mathbf{d}_j \in \Re^3$ of the respective Voronoi polyhedra covering a torus.

The locations of cortical neurons in a sheet suggests that the neuronal connections are biased towards a two-dimensional topology. This is, in fact, the case since the neurons are initially endowed with connections to their neighbours. These connections do not follow the strict pattern of a Delaunay triangulation, rather a single neuron is connected also to next-nearest and next-next-nearest neighbours on so on, and with a connection strength which is not Boolean, but rather reflects a graded strength which, in general favours closer neighbours. The connection (synaptic) strength $C_{jk}$ measures the ability of neuron $j$ to contribute to the excitation of neuron $k$; strong positive values lead to a tendency that neurons $j$ and $k$ tend to be excited together. The connection strengths can also assume negative values, and do so often for neurons within a shell of a certain distance (MARR 1982). This implies that a neuron tends to suppress the activation of neurons in the respective shell. Connection strengths vanish for neurons beyond this shell, but there exist many longer range synapses of key functional importance.

One must note here that the imprinting of a two-dimensional topology onto the cortical neurons is by no means a necessity; actual connections between neurons could represent any dimension, as if a $D$-dimensional Delaunay tesselation is "squeezed" into a two-dimensional sheet with all edges intact. In fact, the two-dimensional connections of neurons are certainly an oversimplification and there is strong evidence that further adjustable, so-called plastic, connections exist and play a role for the shape of the neurons' Voronoi polyhedra (receptive fields). But a predominance of two-dimensional features appears to exist in area v1 of teh visual cortex.

At this point one is faced with a a fascinating dilemma: If cortical neurons in v1 live in a two-dimensional space how can they represent the three-dimensional data torus $T$? The dilemma is more fascinating on account of the fact that the cyclic coordinate $\phi$ describing directional visual features cannot be mapped continuously onto the cortical sheet, a two-dimensional manifold; if such maps are thought, singular points arise near which lie neurons representing rapidly different directions. Due to the cyclic nature of the variable $\phi$ many such points can exist and the number of singular points can be roughly related to how many times the sheet attempts to represent all directions.

Rather than pushing further our theoretical deliberations we seek guidance from observation. Experiments carried out by BLASDEL (1992a,1992b,1994) and GRINVALD (1986) provided information how the directional sensitivity of neurons, e.g., $\phi$, is mapped to area v1. A typical map for the macaque is presented in Fig. 4 (left hand side). The figure presents $\phi$ through a color wheel, the various colors corresponding to different $\phi$-values. A large number of singular points can be recognized, a sample (point 2) being indicated. A better characterization of this map is shown on the right hand side of Fig. 4 which shows the so-called iso-orientation lines, i.e., all points (neurons) in the cortical sheet which are maximally sensitive to a certain orientation. These lines are found to converge into certain points on the sheet, namely the singular points. The part of area v1 shown in Fig. 4 exhibits about fifty singular points and, hence, represents orientations about fifty times.

The map in Fig. 4 includes also the so-called occularity, i.e., the representation of left and right eye. The domains corresponding to the eyes are delineated through thin lines which run nearly orthogonally to the iso-orientation lines. The domains form columns, the so-called occularity columns. Figure 4 does not represent the mapping of locations in v1 to positions in the visual field. This mapping will be discussed further below.

The question arises according to which principles the map in Fig. 4 has been selected by evolution for the primary visual representation. In this respect it should be noted that, in the macaque, visual maps form during the first few months after birth and are driven by visual input, as demonstrated in visual deprivation experiments. One possibility to illucidate the principles behind the distribution of receptive field properties in v1 is to postulate morphogentic rules which reproduce the observed maps in computer simulations and mathematical analyses. This program has been carried out in (OBERMAYER et al. 1992). The work showed that certain rules, which maximize the continuity of the map from the cortical sheet to the data space for $(e, x, y, \phi)$, yield a pattern of receptive

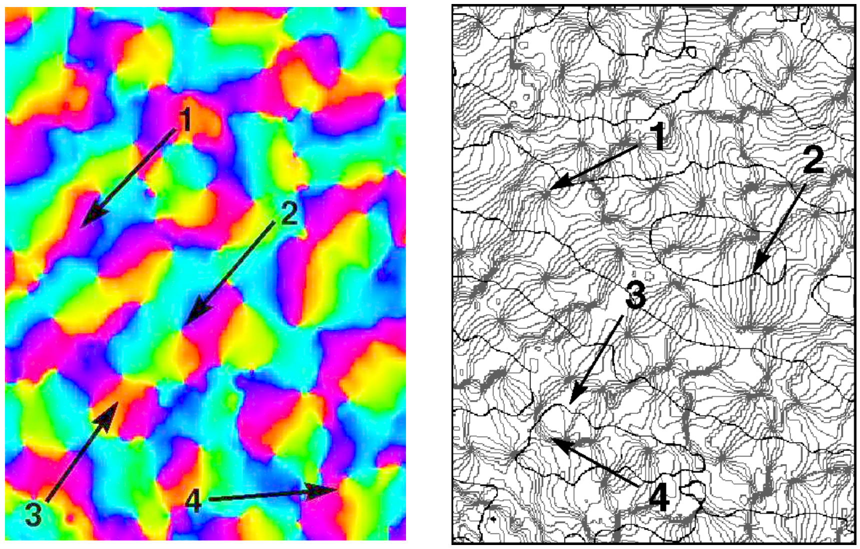


FIG. 4. **Left:** The lateral spatial pattern of orientation preference in the striate cortex of an adult macaque as revealed by optical imaging. The figure (BLASDEL 1992) shows a $4.1\,\mathrm{mm} \times 3.0\,\mathrm{mm}$ surface region located near the border between cortical areas 17 and 18 and close to the midline. (Animal NM1 in (OBERMAYER 1993)) Local average orientation preference is indicated by color such that the interval of 180° is mapped onto a color circle. Arrows indicate (1) linear zones, (2) singularities, (3) saddle points, and (4) fractures. **Right:** Macaque orientation and ocular dominance data combined (OBERMAYER et al. 1992; OBERMAYER and BLASDEL 1993). Black contours separate bands of opposite eye dominance. Light gray iso-orientation contour lines indicate intervals of $11.25°$. The medium gray contour represents the preferred orientation $0°$. Arrows indicate (1) singularities, (2) linear zones, (3) saddle points, and (4) fractures. (ERWIN et al. 1995).

field properties which is in close agreement with neurobiological observation based on voltage sensitive dyes. In (OBERMAYER et al. 1992) both observed maps and model maps are compared and were found in excellent agreement. In (ERWIN et al. (1995)) the various models suggested for the formation of the primary visual representation in the cortex are compared and, in particular, the characteristics of the singular points in the map, as shown in Fig. 4, and similar maps obtained by others are analyzed.

It is of interest to take a closer look at the dynamics of the formation of the visual maps in brains. The character of the map in Fig. 4 is very much determined by the dilemma that the cortical sheet, a two-dimensional manifold, seeks to map into a space of visual features, parametrized by $(x, y, \phi)$ [we neglect presently occularity) which lie in a three-dimensional space . The formation of the map is actually driven by visual experience, i.e., a set of sample data $(x(t), y(t), \phi(t))$, $t = 1, 2, \cdots$. The data represent the local features of optical images. For the following discussion we follow (OBERMAYER et al. 1992) and note that local features in actual images may not exhibit anisotropies which would allow to attribute a direction $\phi$. An example is a painting by the pointillist Seurat who practiced some of his art applying only small color dots to the canvas. In a less extreme case local features might show anisotropic features to various degrees as described by ellipses, characterized by an excentricity $\epsilon$ and the angle $\phi$ of teh major axis. If one would train a young eye–brain system with a pointillistic painting, the cortex would actually not experience anisotropies and actually need to map the cortex sheet into a finite domain in $\Re^2$ which can be realized very easily as discussed in (RITTER et al. (1992), RITTER et al. (1990)). However, re-

alistic scenes will feature local anisotropies which can be characterized through a mean square deviation $\sigma$ into $\epsilon, \phi$–space. The larger $\sigma$ for a set of training images, the more significant is the third dimension and the more the neurons in the cortical sheet seek to represent this dimension. The resulting map can be described through an elastic sheet which is marked by a square grid representing the Cartesian coordinates of the cortical sheet. This sheet is placed into the three-dimensional data space such that cortex point $\vec{\rho}_j$ is pulled to the data point $(x, y, \phi$ stretching the sheet into $\Re^3$). Figure 5 shows a typical result for a case where $\phi$ is assumed to be not periodic. One can recognize that the map protrudes into the third dimension, but at a price of distorting the two-dimensional grid and leading even to folds which imply discontinuities connected with multiple mappings. The dynamics of maps like in Fig. 5 where described in (RITTER andSCHULTEN 1988)) and in the textbooks (RITTER et al. (1992), RITTER et al. (1990)). There it is shown, for example, that the shape of the maps is governed by a phase transition which is governed by the property $\sigma$ of the training images: below a critical $\sigma$-value maps will actually resist to explore the third dimension, exhibiting only reversible fluctuations with amplitudes for certain modes which increase very strongly (singularity) near the critical point. For larger $\sigma$ values the training induces a buckling of the map, as shown in Fig. 5 which is irreversible.
It is fascinating to follow the character of the maps when $\sigma$ is continuously increased such that it eventually exceeds the extension of the $(x, y)$ domain; in this case the hierarchy of the representation becomes inverted and maps, e.g., in case of the visual map in Fig. 4, separate into representations $n = 1, 2, \ldots$ of a set of orientations $\phi_n$, such that each representations keeps $\phi_n$ fixed and fills out the visual coordinates $(x, y)$, i.e., there develops a first map $\vec{\rho}_j \rightarrow (x_j, y_j, \phi_1)$, a second map $\vec{\rho}_j \rightarrow (x_j, y_j, \phi_2)$, etc (OBERMAYER et al. 1992).

## VISUO-MOTOR CONTROL IN ROBOTICS

In this section we want to discuss first how a generalization of the Delaunay triangulation algorithm described above can serve, in principle, to solve the problem of biological visuo-motor control. This approach, followed in (HESSELROTH et al. 1993) and (SARKAR and SCHULTEN), will be outlined first. We describe then an algorithm which is modelled in closer analogy to biological motor control and discuss its application to a *SoftArm* robot system.

### The *SoftArm* Robot System

Movement of higher biological organisms is the result of information processing in a complex hierarchy of motor centers within the nervous system. To date, there is still no general consensus about how biological neural networks actually generate voluntary movement. Neurophysiological studies, on one side, provide the essential data on which a top down modelling approach can be based. On the other side, there is the engineering discipline of robotics which seeks to design robust and adaptible robotic systems, often under the perspective of a specific task. In contrast to biology, robotic control applications based on artificial neural networks are, to a large extent, still confined to systems capable of performing simple sensory-to-motor transformations. Here we seek to combine both sides, engineering and biology, while focusing on the implementation of biologically motivated neural algorithms on a pneumatically driven robot arm (*SoftArm*). The *SoftArm*'s hysteretic behavior makes this arm difficult to control by conventional methods with the

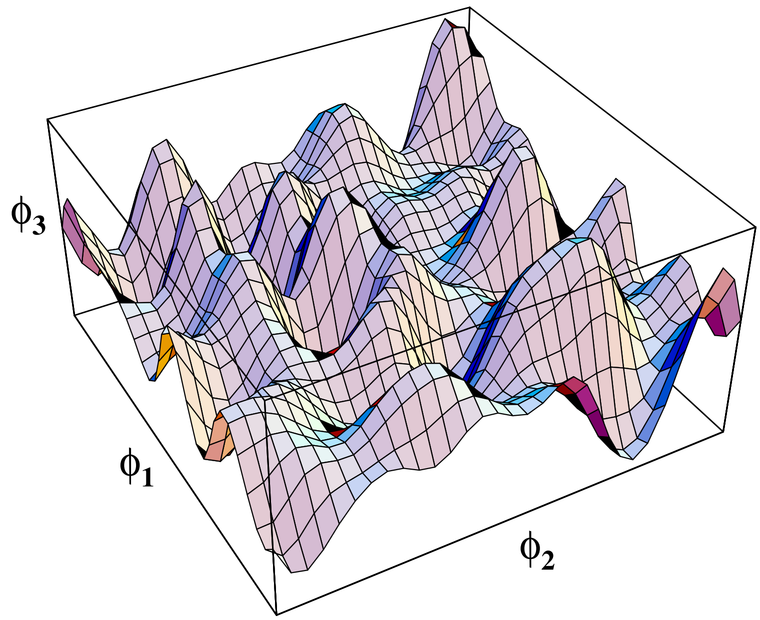


FIG. 5. Dimension-reduction: This figure shows how points in a two-dimensional array might be mapped into a three-dimensional feature space with components $\phi_1$, $\phi_2$ and $\phi_3$, representing such features as visual field location and ocular dominance. Dimension-reduction models often constrain the map to fill the input space with near-uniform density while maintaining continuity. This leads to maps where rapid changes in one feature vector component are correlated with slow changes in other vector components. (ERWIN et al. 1995).

accuracy needed for real-world applications. On the other hand, its unique physical flexibility is a very desirable quality in many applications, such as various human-robot interaction scenarios.

The *SoftArm* is modeled after the human arm and has four joints resulting in five degrees of freedom. It exhibits the essential mechanical characteristics of skeletal muscle systems employing agonist-antagonist pairs of *rubbertuators* which are mounted on opposite sides of rotating joints (see Figure 6). When air pressure in a *rubbertuator* is increased, the diameter of the tube increases, thereby, causing the length of the tube to decrease and the joint to rotate. The motion of each joint $j$, hence, is controlled by two pressure variables of the corresponding pair of tubes (see Figure 6), the average pressure $\bar{p}_j$ in the two tubes and the pressure difference $\Delta p_j$ between the two tubes. Pressure difference drives the joints, average pressure controls the force (complience) with which the motion is executed. This latter feature allows operation at low average pressures and, thereby, allows one to carry out a complient motion of the arm. This makes such robots suitable for operation in a fragile environement, in particular, allows direct contact with human operators. The price to be paid for this advantage is that the response of the arm to signals $(\bar{p}_1, \bar{p}_2, \ldots, \bar{p}_N)^T$ and$(\Delta p_1, \Delta p_2, \ldots, \Delta p_N)^T$ cannot be described by 'a priori' mathematical equations, but rater must be acquired heuristically.

One expects that the response characteristics change during the life time of the arm through wear, after replacement of parts and, in particular, through hysteretic effects. The hysteretic behavior of the arm is demonstrated in Figure 7 which shows a slow relaxation to a steady state, hysteresis in positioning and a change in the pressure-position relationship over time (long-term drift). In consequence, accurate positioning of the *SoftArm* presents a challenging problem. For a more

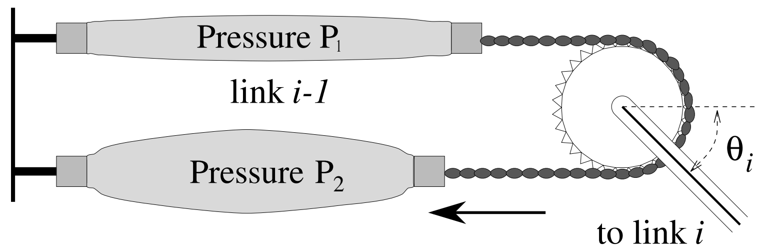


FIG. 6. Agonist and antagonist rubbertuator are connected via a chain across a sprocket. Their relative lengths determine the joint position $\theta_i$, while the sum of the pressures $P_1 + P_2$ modulates the joint stiffness.

detailed introduction to the mechanics of the *SoftArm* see HESSELROTH et al. (1994) .
Figure 8 illustrates the complete robot system, including the mechanical side (*SoftArm*, air supply, servo valve units), the interface (servo drive units) and the controlling workstation. The host computer, currently a Hewlett Packard HP755/99, includes the hardware components (terminal server interface, video input card) and the software layer (robot control library, neural network and image processing program).
Visual feedback is provided by color video cameras. For maximum flexibility, vision preprocessing is implemented in software rather than in hardware. A flow chart of the video stream is illustrated in Figure 9. The use of a frame grabber to import the video signals in a JPEG encoded format minimizes the amount of data to be transferred between the video board and workstation memory. The location of the gripper is extracted from the video frames through a simple color separation, yielding one color component. This is then thresholded and the center of mass of the remaining image calculated. Coding the gripper in a certain color, e.g. red, allows us to weaken the workspace scenery restrictions in terms of background and lighting conditions while, at the same time, keeping the visual preprocessing as simple and efficient as possible.
The *SoftArm* can be operated either in position control mode or pressure control mode. In position control mode, the servo drive units use the joint angle information, provided by optical encoders attached to the joints, and a classical feedback loop to control the actual position of the arm. The control is rather rudimentary and achieves only a low accuracy, mainly limited by the above mentioned mechanical characteristics.
A satisfactory application of the *SoftArm* requires an adaptive control mechanism which can overcome the nonlinear and hysteretic mechanical limitations. The algorithm we seek, in its simplest ramification, should move the gripper at the end of the arm, the so-called end effector, to specified positions $\mathbf{v}$ in the robot's three-dimensional work space $V$. In its simplest form, the $N$ angles $\theta_1, \theta_2, \ldots, \theta_N$ at a robot's joints (we assume presently that $N$ is variable) need to be specified as to achieve the desired position $\mathbf{v}$ of the arms gripper[2]. For this purpose one needs to learn the vector-valued function $\mathbf{v}(\vec{\theta})$, where $\vec{\theta}$ represents the $N$-dimensional column vector $(\theta_1, \theta_2, \ldots, \theta_N)^T$. In case of $N, = 3$, the functional dependence represented by $\mathbf{v}(\vec{\theta})$ is unique (actually, for wide intervals from which angles $\theta_j$ can be taken, the function can assume two or more discrete branches), for $N > 3$ a continuum of possibilities exists to realize end effector positions $\mathbf{v}$. In the latter case one usually wants to select $\theta$ such that certain conditions are met, e.g., that the arm reaches around obstacles. The issues are discussed at length in (RITTER et al. (1992), RITTER et al. (1990)).

---

[2]We will not be concerned at present with the need to properly orient the gripper to grasp an object.

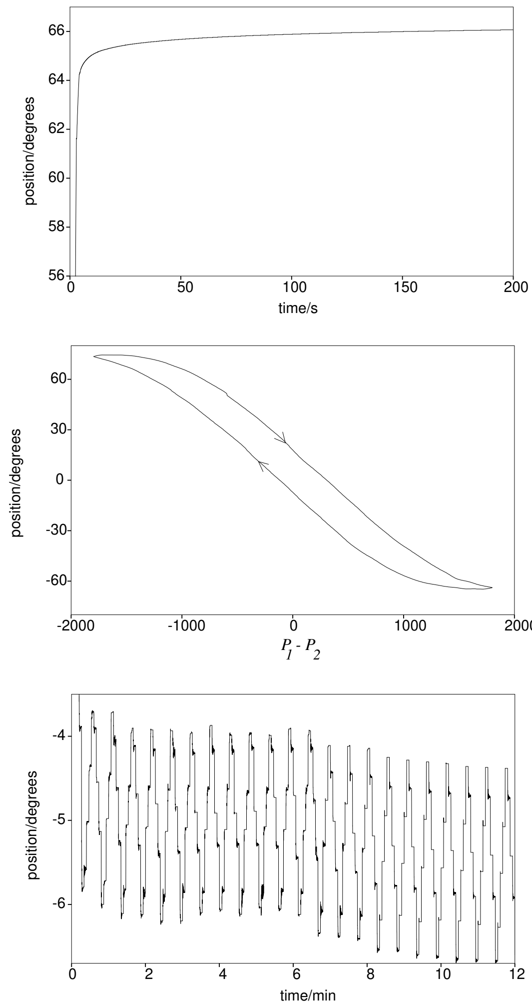


FIG. 7. Mechanical characteristics of the *SoftArm's* joint 1, caused by the use of *rubbertuators*: (top) Slow relaxation to the final position in pressure control mode. (middle) Hysteresis when alternating between two pressure vectors and applying a constant pressure increment/decrement $\Delta P$ to rubbertuator 1 and 2. When the extreme pressures are reached, the direction is reversed. (bottom) Long-time drift of the position while repeatedly changing the total pressure by $\pm 1\%$. (HESSELROTH et al. 1994).

The control problem just stated can be solved by conventional robot algorithms. The situation becomes more difficult, and more interesting, in case that the control signals actually employed do not specify directly the joint angles, as in case of the *SoftArm* in pressure mode. How can one obtain information on the response characteristics of the robot arm? We have suggested earlier (RITTER et al. (1992), RITTER et al. (1990)) to employ a pair of stereo cameras. We have demonstrated in conjunction with an industrial robot ( WALTER) and the *SoftArm* system (HESSELROTH et al. 1993) , (SARKAR and SCHULTEN) that the signals from the two camera backplanes can be employed for this purpose, i.e., the robot-camera-computer system can learn, in fact, to control the arm solely on account of camera images.

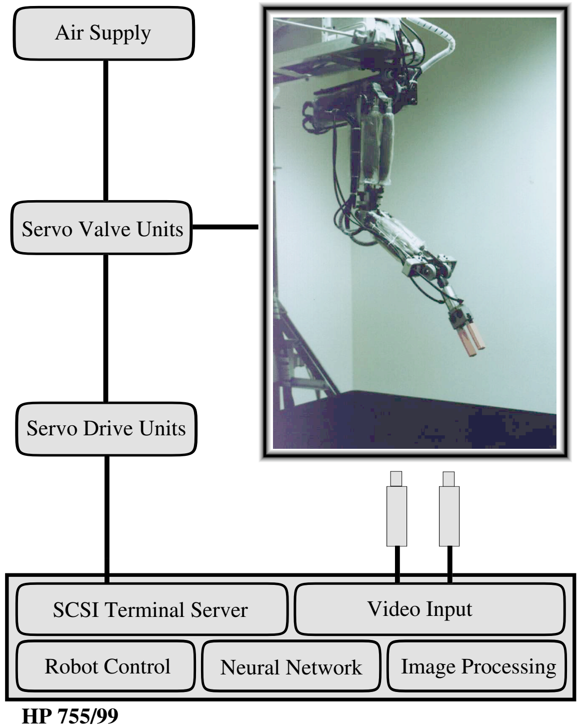


FIG. 8. Diagram of the robot system, showing *SoftArm*, air supply, control electronics and workstation. The host computer includes a software layer (robot control, neural network and image processing programs) and the hardware components (serial interface and video input).

**Topology Representing Network Algorithm for Visuo-Motor Control**

Before we embark on specifying how the topology representing networks can be trained to control robot motion we need to introduce a concept of utmost practical importance, the linearly controlled feed-back loop. The idea is that rather than to learn directly the precise relationship between joint angles and end effector positions, one learns such relationship only approximately and only for a very coarse set $\{\mathbf{v}_s,\, s \in A\}$ of end-effector positions, i.e., one learns a set of joint angles $\vec{\theta}_s,\, s \in A$ for some set $A$ (to be specified later) such that[3]

$$\mathbf{v}_s \;=\; \mathbf{v}(\vec{\theta}_s)\;. \tag{2}$$

The remaining control is assigned to linear feed-back loops which are based on the iteration

$$\mathbf{v}^{(n)} \;=\; \mathbf{v}\left(\vec{\theta}(n-1) \,+\, \mathrm{A}_s\left(\mathbf{v}_{\text{target}} \,-\, \mathbf{v}^{(n-1)}\right)\right) \tag{3}$$

where $\mathrm{A}_s$ is the Jacobian tensor $\partial\vec{\theta}/\partial\mathbf{v}$ evaluated at the locations $\vec{\theta}_s,\, s \in A$. $\vec{\theta}(n-1)$ represents the joint angles after $n-1$ iterations defined through $\mathbf{v}^{(n-1)} \;=\; \mathbf{v}[\vec{\theta}(n-1)]$. This expansion

[3]The function $\mathbf{v}(\cdots)$ specifies the relationship between the joint angles of the robot arm and the position of its end-effector in the work space; this function is determined through the geometry of the robot arm and is available through the actual operation of the robot in joint angle mode: a controller specifies $\vec{\theta}_s$ and the arm moves into a certain position. Likewise, one may use pressures to control the robot operating in pressure mode. The latter mode is actually employed in most applications.

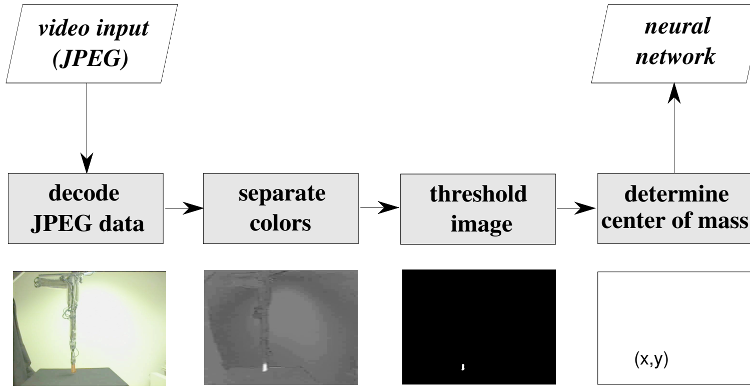


FIG. 9. Flow chart of the vision preprocessing: The true color video input, coming as JPEG encoded data from a Parallax Graphics PowerVideo 700Plus video board, is decoded. After separating the colors (to extract, e.g., the red components) and thresholding the image, the gripper's location is determined by calculating the center of mass for the remaining pixels.

attempts to move the end effector to the target location $\mathbf{v}_{\mathrm{target}}$ by linearly correcting the joint angles on account of the remaining deviation $\mathbf{v}_{\mathrm{target}} - \mathbf{v}^{(n-1)}$. Repeated application of (3), starting with $\mathbf{v}^{(0)} = \mathbf{v}(\vec{\theta}_s)$, leads to a series of end effector positions $\mathbf{v}^{(1)}$, $\mathbf{v}^{(1)}$, $\ldots$ which approaches $\mathbf{v}_{\mathrm{target}}$ for suitable $\mathrm{A}_s$. Schemes for acquiring $\vec{\theta}_s$ and $\mathrm{A}_s$ have been presented in (RITTER et al. (1992), RITTER et al. (1990)) and their capacity for real applications has been demonstrated in ( WALTER). The choice of mesh points (centers of Voronoi Polyhedra) is the most essential part of the algorithm. As explained above, there are two aspects involved, the development of a Voronoi diagram and the development of the associated Delaunay tesselation. The conrol algorithm considered here actually generates, in a training period, a table look-up program. Equations (2, 3) concern actually the table entries. They state that each Voronoi polyhedron, labelled $s$, will be associated with a vector $\vec{\theta}_s$ which represents the angles specifying, through the joint angles, the zero order conformation of the robot arm. However, each polyhedron stores also a linear correction scheme, specified through the tensor $\mathrm{A}_s$; the latter scheme allows approach to a target point $\mathbf{v}_{\mathrm{target}}$ which deviates from the point $\mathbf{v}(\vec{\theta}_s)$. One can consider the present approach an exercise in triangulation for function approximation, except that the function is an iterative process.

It is now quite obvious how the mesh points are chosen. In a training period one issues requests to the robot to move to target points $\mathbf{v}_{\mathrm{target}}(t)$, $t = 1, 2, \ldots$. This training set yields then a Voronoi diagram with associated Delaunay tesselation as outlined above. The difference to the earlier algorithm is that the system establishes at the same time the table entries $\vec{\theta}_s$ and $\mathrm{A}_s$. However, in learning the table entries the Delaunay triangulation comes to play a cardinal role. This role arises from the justifiable expectation that neighbouring Voronoi polyhedra $s$ and $s'$ should assume eventually similar table entries ($\vec{\theta}_s$, $\mathrm{A}_s$) and ($\vec{\theta}_{s'}$, $\mathrm{A}_{s'}$). This can be exploited through cooperation of neighboring polyhedra in acquiring the proper table entries. This cooperation extends initially over next neighbours, next-next nearest neighbours, etc and involves a significant spill over of entry updates from one polyhedron to others. However, in the course of the training, the cooperation becomes more narrowly focussed to immediate neighbours and also involves gradually less of a spill over of entries. Finally, units learn only individually, fine tuning only their own entries. This cooperation furnished through the Delaunay tesselation does not only speed up the training, since each training step during the early phase of the training involves many Voronoi cells, but it also increases the radius of convergence of the algorithm dramatically. This radius is defined by the

initial entries ($\vec{\theta}_s(t=0)$, $\mathrm{A}_s(t=0)$) of the Voronoi cells; if the entries are too far off, the system may never find good ($\vec{\theta}_s$, $\mathrm{A}_s$) entries; cooperation ameliorates the effect of a few cells which may never acquire proper table entries if they would rely solely on their own (poor) initial ($\vec{\theta}_s(t=0)$, $\mathrm{A}_s(t=0)$) entries; however, they acquire through cooperative learning better table entries from their neighbours and the schemes for acquiring proper table entries converges fro them as well.
The Delaunay tesselation can play also an essential role after training is completed. One can envoke cooperation between neighbouring Voronoi cells to average the motor response. This is useful for biological neurons, the signals of which are limited in accuracy to few bits. Averaging can effectively reduce the error of control signals by a factor of about $1/\sqrt{n}$ where $n$ is the number of participating units.

### A Biologically Inspired Control Algorithm

Instead of the formal approach outlined above we consider now a simpler algorithm for linking visual input to motor output, which is inspired by neurobiological observation. This approach involves the application of *self-organizing feature maps*, originally proposed by KOHONEN (1982), as the basic information processing element from which neural networks capable of visuo-motor control are built. Such networks have been successfully applied to the problem of controlling movement in several technical applications (RITTER and SCHULTEN 1986, RITTER et al. 1989, MARTINETZ et al. 1990, COITON et al. 1991, WALTER and SCHULTEN 1993). In common with a number of previous studies of motor control (for example that of KUPERSTEIN (1988)), our present approach involves the development of connections between an input (sensory) and output (motor) map, the connections between these maps being achieved by means of a learning process. In this regard, the study of COITON et al. (1991) is of particular interest, as it extends one general approach suggested by our group (RITTER et al. 1989). COITON et al. define an architecture that learns to control movement *through associations between two sensory modalities*. In the model each neuron within the network receives both exteroceptive input regarding the visual scene and proprioceptive input indicating the instantaneous angular positions of the limb segments of the movement system (e.g. a two jointed planar simulation of the human arm or industrial robotic manipulator). Through random exploration of the workspace, similar to the manner in which immature primates perform apparently random motor acts, neurons within the network are able to develop associations between these different sensory modalities.
From a biological perspective such an approach is appealing, because of the ability of the model to fuse different types of sensory input regarding movement during calculation of the required sequence of motor commands. Such an ability is frequently cited as a principal reason for the superiority of biological control systems when compared to artificial movement systems. To evaluate the success of such a strategy when applied to the problem of accurate positioning of an end-effector we have employed the basic approach outlined by Coiton et.al. to control our *SoftArm*.

### Neural Control of the *SoftArm*

To simplify the description of the algorithm, we will first discuss application of the algorithm for use in the control of 3 joints and a fixed compliance. Figure 10 illustrates the elements of the basic network. Neurons in layer $S$ project *via independent excitatory synapses* to a set of motor cells

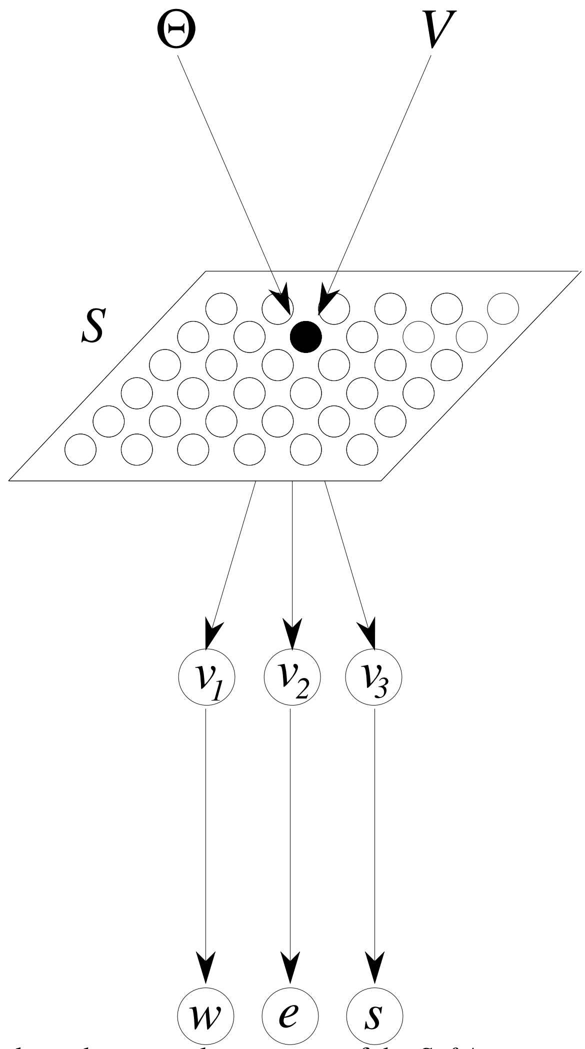


FIG. 10. The basic network used to control movement of the *SoftArm*. $w$, $e$ and $s$, refer to the joint wrist, elbow and shoulder joints of the arm respectively. Definitions of all other quantities may be found in the text.

$v_i$ responsible for setting the pressure values of the joints. We assume an input space defined by M independent sensory input sources. In a biological system these sources might, for example, correspond to neurons providing tactile input from receptors distributed over the body surface. In the present work, however, we will be concerned with proprioceptors, which indicate the respective joint angles of each of the segments of the robot arm, as provided by optical encoders mounted at each joint, and visual receptors which specify the location of a target point of interest within each camera plane.
Hence, two different types of sensory information converge upon neurons within the network $S$. Exteroceptive input

$$\mathbf{r} = [x_1, x_2, x_3, x_4] \tag{4}$$

is derived from a Euclidean coordinate system defined by the normalized visual fields presented to the network. Proprioceptive input, denoted $\mathbf{\Theta}$, is derived from the intrinsic coordinate system of the joints, given by the normalized values of the optical encoders, where

$$\mathbf{\Theta} = [\theta_1, \theta_2, \theta_3] \tag{5}$$

The indices $1, 2, 3$ specify the wrist, elbow and shoulder joints of the *SoftArm*, respectively.
Control of limb movements in the workspace is achieved by modifying the synaptic weights of the projections from neurons in the sensory layer $S$ to the motor cells. Each neuron in the sensory layer has a vector

$$\mathbf{V}_j = [v_j^1, v_j^2, v_j^3] \tag{6}$$

associated with it which corresponds to the output of the motor neuron when activated by a neuron in the sensory layer. This output alters the joint angles by sending new pressure values to the *SoftArm*.
During learning, adjustment of $v_j^i$, the $i^{th}$ component of $\mathbf{V}$, is calculated as

$$v_j^i(t) = v_j^i(t-1) + \epsilon(t) h_{js}(u^i(t) - v_j^i(t-1)) \tag{7}$$

where, in this instance, $v_j^i(t-1)$ represents a *random* value generated during the previous iteration of the algorithm leading to movement of a particular limb segment (COITON et al. 1991). $\epsilon(t)$ determines the magnitude of change in the synaptic weights as a function of time and is chosen in the following manner (RITTER et al. 1992)

$$\epsilon(t) = \epsilon_{ini}(\epsilon_{fin}/\epsilon_{ini})^{t/t_{max}} \tag{8}$$

The neighborhood function $h_{js}$ can be modeled by a Gaussian

$$h_{js} = \exp(- \parallel j - s \parallel^2 / 2\sigma(t)^2) \tag{9}$$

with width

$$\sigma(t) = \sigma_{ini}(\sigma_{fin}/\sigma_{ini})^{t/t_{max}} \tag{10}$$

Prior to learning, all components of the vector $\mathbf{V}$ are assigned random values and the total number of learning steps is specified. For each learning step a sensory input vector $\mathbf{U} = [\mathbf{r}, \mathbf{\Theta}]$ is then formed from exteroceptive input $\mathbf{r}$ given by the values of the endpoint of the limb and proprioceptive input $\mathbf{\Theta}$ specified by the joint angles of the limb. The Kohonen algorithm is applied to the sensory layer and the vector of motor signals $\mathbf{V}_s$ associated with the neuron $s$, chosen according to the Euclidean distance criteria proposed by KOHONEN (1982) , in the sensory layer, initiates movement of the arm to a new randomly chosen position in the workspace. The components of $\mathbf{V}_s$ are then adjusted according to (7) and this sequence of operations is repeated for the total number of learning steps.
Following a suitable number of learning steps, typically 3000, goal-directed movements to visual targets can be executed by the network in the following manner. The limb assumes an initial configuration of joint angles $\mathbf{\Theta}$ corresponding to a position $\mathbf{r}$ of the endpoint of the limb in the workspace. A sensory vector $\mathbf{U}$, concatenated from the Cartesian coordinates of the target position $\mathbf{r}_p$ and the current joint angles of the limb, induces excitation of a neuron $s$ in the sensory layer. This sensory vector codes two physical locations in the workspace: the target location $\mathbf{r}_p$ and the current position of the limb $\mathbf{r}$ as a function of $\mathbf{\Theta}$. Excitation of neuron $s$ in the sensory layer results in a new motor vector $\mathbf{V}_s$ being sent to the limb simulation causing movement of the endpoint of

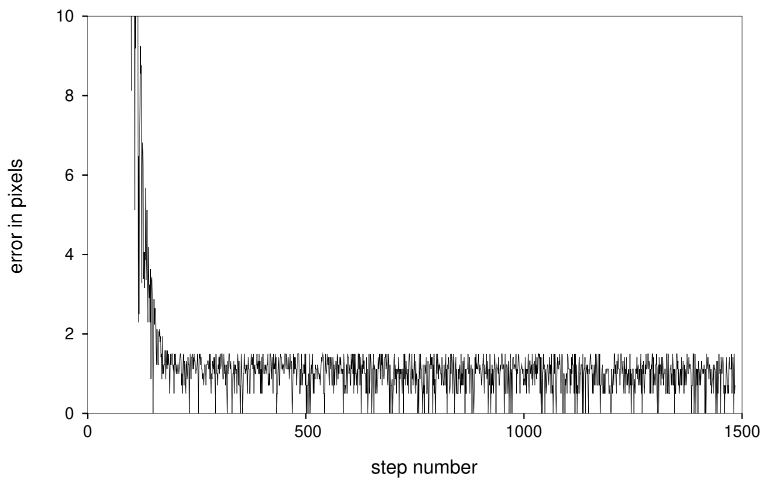


FIG. 11. Positioning error between gripper and target versus step number using the "*neural gas*" algorithm combined with an interpolation strategy. One pixel corresponds to approx. $3mm$. (HESSELROTH et al. 1994).

the limb to a new position $\mathbf{r}_s$ corresponding to the set of joint angles $\boldsymbol{\Theta_s}$. A new vector $\mathbf{U}'$ is then concatenated from $\mathbf{r}_p$ and $\boldsymbol{\Theta_s}$ inducing excitation of neuron $s'$ in the sensory layer. The associated motor vector $\mathbf{V}_{s'}$ then causes movement of the limb to the position $\mathbf{r}_{s'}$ associated with $\boldsymbol{\Theta_{s'}}$. This process continues until the sequence of positions $\mathbf{r}, \mathbf{r}_s, \mathbf{r}_{s'}, \mathbf{r}_{s''}, \ldots$ attained by the endpoint of the limb during movement to the target point converges within a predefined tolerance to $\mathbf{r}_p$, or the same neuron is chosen in two consecutive iterations of the process, or the total number of iterations of the process required by the movement exceeded a predetermined number, typically ten. In general, movement to a particular target point will involve a total of three or four iterations of this sequence of steps, though greater numbers are possible.

Use of this algorithm for control of the *SoftArm* results in average accuracies in the region of 12% of the dimensions of the workspace of this system. A number of factors contribute to this poor performance, including the mechanical characteristics of the *SoftArm* and the high dimensionality of the problem. The principal problems that arise, however, are the need for a single network to provide a 'good' representation of two distinct sensory input spaces, namely the visual and proprioceptive spaces, and the discretizing effect that results from the use of small numbers of neurons to map these input spaces. In general, while the use of larger number of neurons in the network can lead to some improvements in performance there is no simple linear relationship between greater numbers of neurons and accuracy (RITTER 1989).

Through a combination of self-organizing feature maps and interpolation strategies it is possible to overcome the discretizing problem, to achieve a precise end effector position control (HESSELROTH (1994)) and to orient the gripper properly for an object to be grasped (SARKAR and SCHULTEN). In addition to the *neural gas* algorithm, representing the three-dimensional workspace, the robot also learns a set of Jacobian matrices for interpolating between positions stored by 200 individual neurons. In Figure 11 we plot the final error between gripper and target position (in camera pixels) versus the number of learning steps.

### Future Directions in Visuo-Motor Control

Biological organisms provide both, the inspiration and the challenge for robotic systems. They are the basis on which industrial robotic applications, controlled by artificial neural networks, have to be evaluated. The *SoftArm*, with it's highly nonlinear and hysteretic characteristics, represents an interesting and challenging experiment for those neural network architectures. In addition to the positioning we demonstrated here, more complex tasks, e.g. grasping objects of arbitrary shape, motion path planning or tracking of moving targets are currently the focus of research efforts. However, the much more sophisticated capabilities of biological organisms in terms of visuo-motor control suggest to emphasize the biological design of the implemented neural networks. A synergy between the engineering approach of robotics and biologically motivated models of motor learning will, on one side, lead to a better performance in robotics and, on the other side, elucidate the manner in which the various motor centers within the cerebral cortex jointly program and coordinate movement.

## ACKNOWLEDGMENTS

The authors like to thank Ed Erwin, Thomas Martinetz, Klaus Obermayer, and Ken Wallace for a fruitful collaboration on the work described here. This work was supported by the Carver Charitable Trust.

## LITERATURE

G.G. BLASDEL. Differential imaging of ocular dominance and orientation selectivity in monkey striate cortex. *J. Neurosci.*, 12(8):3139–3161, 1992.

G.G. BLASDEL, K. OBERMAYER, and L. KIORPES. Organization of ocular dominance and orientation columns in the striate cortex of neonatal macaque monkeys. *Vis. Neurosci.*, In press.

G.G. BLASDEL. Orientation selectivity, sreference and continuity in monkey striate cortex. *J. Neurosci.*, 12(8):3115–3138, 1992.

BOLLOBÁS. *Graph Theory, An Introductory Course*. Springer-Verlag, New York, 1979.

J. BRAUN and M. SAMBRIDGE. A numerical method for solving partial differential equations on highly irregular evolving grids. *Nature*, 376:655–660, 1995.

Y. COITON, J.C. GILHODES, J.L. VELAY, and J.P. ROLL. A neural network model for the intersensory coordination involved in goal-directed movements. *Biol. Cybern.*, 66:167–176, 1991.

E.W. DIJKSTRA. A note on two problems in connection with graphs. *Numer. Math.*, 1(5):269–271, 1959.

E. ERWIN, K. OBERMAYER, and K. SCHULTEN. Models of orientation and ocular dominance columns in the visual cortex: A critical comparison. *Neural Computation*, 7:425–468, 1995.

J.A. GEORGE. Computer implementation of the finite element method. Technical Report STAN-CS-71-208, Computer Science Department, Stanford University, 1971.

J.C. GOWER and G.J.S. ROSS. Minimum spanning trees and single linkage cluster analysis. *Appl. Stat.*, 18(1):54–64, 1969.

R.M. GRAY. Vector quantization. *IEEE ASSP*, 1(2):4–29, 1984.

A. GRINVALD, E. LIEKE, R. P. FROSTIG, C. GILBERT, and T. WIESEL. Functional architecture of cortex revealed by optical imaging of intrinsic signals. *Nature*, 324:351–354, 1986.

T. HESSELROTH, K. SARKAR, P. VAN DER SMAGT, and K. SCHULTEN. Neural network control of a pneumatic robot arm. *IEEE Transactions of System, Man and Cybernetics*, 24(1):28–37, 1994.

D. HUBEL and T.N. WIESEL. Receptive fields, binocular interaction and functional architecture in the cat's striate cortex. *J. Physiol. (Lond.)*, 160:106–154, 1962.

D.E. KNUTH. *The Art of Computer Programming. Volume III: Sorting and Searching*. Addison-Wesley, Reading, MA, 1973.

T. KOHONEN. Analysis of a simple self-organizing process. *Biol. Cybern.*, 44:135–140, 1982.

T. KOHONEN, K. MÄKISARA, and T. SARAMÄKI. Phonotopic maps - insightful representation of phonological features for speech recognition. In *Proc. 7th Int. Conf. on Pattern Recognition, Montreal*, pages 182–185, 1984.

J.B. KRUSKAL. On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proc. AMS 7*, pages 48–50, 1956.

M. KUPERSTEIN. Neural model of adaptive hand-eye coordination for single postures. *Science Wash. DC*, 239:1308–1311, 1989.

D. LEE and J. MALPELI. Global form ans singularity: Modeling the blind spot's role in lateral geniculate morphogenesis. *Science*, 263:1292–1294, 1994.

J. MAKHOUL, S. ROUCOS, and H. GISH. Vector quantization in speech coding. *Proc. IEEE*, 73:1551–1588, 1982.

D. MARR. *Vision*. Freeman, San Francisco, 1982.

T. MARTINETZ, S.G. BERKOVICH, and K. SCHULTEN. Neural-gas network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4(4):558–569, 1993.

T. MARTINETZ, H. RITTER, and K. SCHULTEN. Three-dimensional neural net for learning visuo-motor coordination of a robot arm. *IEEE Transactions on Neural Networks*, 1:131–136, 1990.

T. MARTINETZ and K. SCHULTEN. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.

N.M. NASRABADI and Y. FENG. Vector quantization of images based upon the kohonen self-organizing feature maps. In *IEEE International Conference on Neural Networks, San Diego*, pages 1101–1108, 1988.

N.M. NASRABADI and R.A. KING. Image coding using vector quantization: A review. *IEEE Trans. Comm.*, 36(8):957–971, 1988.

J. NAYLOR and K.P. LI. Analysis of a neural network algorithm for vector quantization of speech parameters. In *Proc. of the First Annual INNS Meeting*, page 310, New York, 1988. Pergamon

Press.

K. OBERMAYER. *Adaptive Neuronale Netze und ihre Anwendung als Modelle der Entwicklung Kortikaler Karten*. Infix-Verlag, St. Augustin, 1993.

K. OBERMAYER and G. BLASDEL. Geometry of orientation and ocular dominance columns in monkey striate cortex. *J. Neurosci.*, 13:4114–4129, 1993.

K. OBERMAYER, G.G. BLASDEL, and K. SCHULTEN. Geometry of orientation and ocular dominance columns in monkey striate cortex. *Physical Review A*, 45(10):7568–7589, 1992.

K. OBERMAYER, G.G. BLASDEL, and K. SCHULTEN. Statistical-mechanical analysis of self-organization and pattern formation during the development of visual maps. *Physical Review A*, 45(10):7568–7589, May 1992.

S.M. OMOHUNDRO. The delaunay triangulation and function learning. Technical Report TR-90-001, International Computer Science Institute, Berkeley, CA, 1990.

R.E. OSTEEN and P.P. LIN. Picture skeletons based on eccentricities of points of minimum spanning trees. *SIAM J. Comput.*, 3(1):23–40, 1974.

F.P. PREPARATA and M.I. SHAMOS. *Computational Geometry: An introduction*. Springer-Verlag, New York, 1985.

R.C. PRIM. Shortest connecting networks and some generalizations. *BSTJ*, 36:1389–1401, 1957.

H. RITTER. Combining self-organizing maps. In *International Joint Conference on Neural Networks*, 1989.

H. RITTER, T. MARTINETZ, and K. SCHULTEN. *Neural Computation and Self-Organizing Maps*. Addison-Wesley, New York, 1992.

H. RITTER, T. MARTINETZ, and K. SCHULTEN. *Textbook: Neuronale Netze: Eine Einführung in die Neuroinformatik selbstorganisierender Abbildungen*. Addison-Wesley, Bonn, first edition, 1990.

H. J. RITTER, T. M. MARTINETZ, and K. J. SCHULTEN. Topology-conserving maps for learning visuo-motor-coordination. *Neural Networks*, 2:159–168, 1989.

H. RITTER and K. SCHULTEN. Topology conserving mappings for learning motor tasks. In J. S. DENKER, editor, *Neural Networks for Computing*, pages 376–380. American Institute of Physics Publication, Conference Proceedings 151, 1986.

H. RITTER and K. SCHULTEN. Convergence properties of Kohonen's topology conserving maps: Fluctuations, stability and dimension selection. *Biological Cybernetics*, 60(1):59–71, 1988.

D.J. ROSENKRANTZ, R.E. STEARNS, and P.M. LEWIS. Approximate algorithms for the traveling salesperson problem. In *Fifteenth Annual IEEE Symposium on Switching and Automata Theory*, pages 33–42, 1974.

J. RUBNER and K. SCHULTEN. Development of feature detectors by self-organization: A network model. *Biological Cybernetics*, 62:193–199, 1990.

J. RUBNER, K. SCHULTEN and PAUL TAVAN. A self-organizing network for complete feature extraction. In R. ECKMILLER, G. HARTMANN, and G. HAUSKE, editors, *Parallel Processing in*

*Neural Systems and Computers*, pages 365–368. Elsevier, Amsterdam, 1990.

K. SARKAR and K. SCHULTEN. Topology representing network in robotics. In J.L. VAN HEMMEN, E. DOMANY, and K. SCHULTEN, editors, *Physics of Neural Networks, Volume 3*. Springer–Verlag, New York, 1995. In press.

M.I. SHAMOS. *Computational Geometry (PhD thesis)*. Dept. of Computer Science, Yale Univ., 1978.

G. STRANG and G. FIX. *An analysis of the finite element Method*. Prentice-Hall, Cliffs, NJ, 1973.

S. TZONEV, J. MALPELI, and K. SCHULTEN. Morphogenesis of the lateral geniculate nucleus: How singularities affect global structure. In G. TESAURO, D. TOURETZKY, and T. LEEN, editors, *Advances in Neural Information Processing Systems 7*, pages 133–140, Cambridge, Mass and London, England, 1995. MIT Press.

J.A. WALTER and K. SCHULTEN. Implementation of self-organizing neural networks for visuo-motor control of an industrial robot. *IEEE Transactions on Neural Networks*, 4(1):86–95, 1993.

C.T. ZAHN. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. Comp.*, C-20(1):68–86, 1971.