Canonical Parameterization of Excess Motor Degrees of Freedom with Self-Organizing Maps

David DeMers and Kenneth Kreutz-Delgado, *Senior Member, IEEE*

***Abstract*—The problem of sensorimotor control is underdetermined due to excess (or "redundant") degrees of freedom when there are more joint variables than the minimum needed for positioning an end-effector. A method is presented for solving the nonlinear inverse kinematics problem for a redundant manipulator by learning a natural parameterization of the inverse solution manifolds with self-organizing maps. The parameterization approximates the topological structure of the joint space, which is that of a fiber bundle. The fibers represent the "self-motion manifolds" along which the manipulator can change configuration while keeping the end-effector at a fixed location. The method is demonstrated for the case of the redundant planar manipulator. Data samples along the self-motion manifolds are selected from a large set of measured input–output data. This is done by taking points in the joint space corresponding to end-effector locations near "query points," which define small neighborhoods in the end-effector work space. Self-organizing maps are used to construct an approximate parameterization of each manifold which is consistent for all of the query points. The resulting parameterization is used to augment the overall kinematics map so that it is locally invertible. Joint-angle and end-effector position data, along with the learned parameterizations, are used to train neural networks to approximate direct inverse functions.**

I. Introduction

THE forward kinematics function of a manipulator

$$x = f(\theta) \tag{1}$$

maps a set of joint values $\theta \in \Theta$, a configuration, to an end-effector location in the reachable workspace, $x \in \mathcal{W} \triangleq f(\Theta)$. It is assumed that the dimensionality of the configuration space Θ (also called the joint space) exceeds that of $\mathcal{W}$, in which case the manipulator is said to be redundant.

Tasks are specified in the workspace, necessitating the solution of the inverse problem

$$\text{Given target } x_d \in \mathcal{W}, \text{ find } \theta \in \Theta \text{ such that } f(\theta) = x_d. \tag{2}$$

This inverse problem is ill-posed; the inverse or preimage, $f^{-1}(x_d)$, of a given point x_d in the workspace of a redundant manipulator is generically a set of points in the configuration space of dimensionality equal to the number of redundant degrees-of-freedom (dof). As discussed in the next section, this set consists of a finite number of disjoint manifolds[1] for any $f(\cdot)$ which is a smooth map between compact manifolds [1].

Finding a specific configuration θ which positions the end-effector at a given target location requires a selection of one of these disjoint manifolds, and a resolution of the redundant dof by locating a specific point along the manifold. Which point should be chosen out of the infinite set of solution points will likely depend on additional criteria, such as optimizing for various side functionals.

Neural-network based motor control of limbs or manipulators with redundant dof is an important problem of topical interest, relevant to researchers in learning theory, robotics and neurophysiology, [2]–[10]. The research reported in these references can be separated into two general classes of approaches: direct approaches, in which an explicit approximation to f^{-1} is sought, and indirect approaches, which typically use recurrent networks or iterative gradient-based methods to reach a local solution to Problem (2). The indirect methods require recurrent networks or iterative solutions at run-time and result in local optimization of run-time specified side constraints [11], [6].

Finding f^{-1} directly is problematic because the set of points θ which solve $f(\theta) = x_d$ is not unique. The inverse is a one-to-many mapping and thus not a function. We can, however, speak in a restricted sense of inverse functions over a subset of the workspace $\mathcal{W}$ [12]. A continuous function $g(\cdot)$ is an inverse function of $f(\cdot)$ over $\mathcal{U} \subset \mathcal{W}$ if $f(g(x)) = x, \forall x \in \mathcal{U}$. The direct approach to inverse kinematics seeks to approximate an inverse function $g(\cdot)$. As implemented in the past, however, these approaches generally either assume that the kinematics relationship between joint variables and end-effector positioning is one-to-one and onto (i.e., is nonredundant), or else regularize the problem by adding additional constraints at training time (generally with methods which find a locally optimal solution), thereby forcing one-to-one behavior [2], [5]. In the latter case, an inverse function is constructed which makes the extra dof unavailable for satisfying new constraints arising at run-time.

This paper demonstrates that one can learn and parameterize the global topological structure of many-to-one maps and apply direct neural-network methods to the inverse kinematics problem of redundant manipulators without having to enforce

Manuscript received October 13, 1993; revised May 10, 1994 and August 18, 1994. This work was supported by NSF Grant IRI-9202581.

D. DeMers is with the Computer Science and Engineering Department and the Institute for Neural Computation, University of California at San Diego, La Jolla, CA 92093-0114 USA.

K. Kreutz-Delgado is with the Electrical and Computer Engineering Department and the Institute for Neural Computation, University of California at San Diego, La Jolla, CA 92093-0407 USA.

Publisher Item Identifier S 1045-9227(96)00487-0.

[1] A manifold is a space which is locally, but not necessarily globally, Cartesian. For example, the surface of a sphere is a two-dimensional manifold, but cannot be mapped one-to-one to any connected subset of the plane. A compact manifold is closed and bounded.

a task-specific regularization at training time. Effectively, a natural representation of the redundant dof can be learned which parameterizes a family of direct inverse functions. The resulting solution can be flexibly chosen at run-time to meet any redundancy-resolving side task or criterion. The approach is demonstrated on the three-link all rotational joint (3R) planar manipulator.

The remainder of the paper is organized as follows. In Section II, the topology of 3R robot kinematics is reviewed and the trivial fiber bundle structure of the solution branches is explained. In Section III, a brief survey of algorithmic and neural-network approaches to the inverse kinematics problem for redundant manipulators is given, and the self-organizing map (SOM) approach of this paper is first introduced. In Section IV, the method of this paper is further described and a learning algorithm is presented which results in approximating a canonical representation of each solution branch, exploiting its fiber bundle structure. We also discuss the details of an application of our method to the 3R planar manipulator. In Section V, the parameterizations constructed are used to develop regularized direct inverse kinematics functions for the 3R planar manipulator. Finally, Section VI concludes with a summary of the results, a discussion of generalizations to other problems, and a discussion of future research directions.

II. Topological Structure of 3R Kinematics

In this section, we describe the topology of the abstract joint angle space and workspace of a 3R planar manipulator. We refer the reader to [17] and [15] for more detail and specific theoretical results. Our topological analysis applies to smooth functions over compact domains, thus the techniques developed in this paper can be used in a wide variety of inverse problems, not simply for inverse kinematics.

The forward kinematics function of the 3R manipulator considered in this paper is a mapping from a three-dimensional compact set, the three-torus $T^3 = S^1 \times S^1 \times S^1$, ($S^1$ denoting the circle) to a two-dimensional compact set, $\mathcal{W} = f(T^3) \subset R^2$

$$f\colon \theta \in \Theta = T^3 \mapsto x \in \mathcal{W} \subset R^2.$$

The preimage of each end-effector location, $f^{-1}(x)$, is called the fiber over x; it is generically a one-dimensional manifold in the joint angle space [1]. The physical significance of these manifolds is that all points on any one preimage manifold result in the same end-effector position. These are the so-called "self-motion" manifolds [14]. If the configuration of the manipulator follows one of these manifolds, the end-effector location does not move, although the individual links of the manipulator can move.

It is known that, in general, no single continuous inverse function exists for the 3R manipulator which is valid globally over the entire workspace [13]. Continuous inverse functions, however, do exist over certain disjoint singularity-free regions, $\mathcal{W}_i$, of the workspace, namely the so-called w-sheets of [14]. The regions in the joint angle space which are the inverse-images of the w-sheets $\mathcal{W}_i \subset \mathcal{W}$ have a specific topological structure which can be exploited to construct direct inverse functions, as will be shown below.

The workspace of the 3R planar manipulator, $\mathcal{W}$ can be partitioned into either three or four w-sheet regions [15]. The boundaries of the w-sheets are the image in $\mathcal{W}$ of the singularities of $f(\cdot)$. These boundaries are called "critical value surfaces" [14], or "Jacobian surfaces" [16]. Fig. 1 illustrates the w-sheet partition of the workspace and the corresponding inverse regions in the configuration space for the 3R planar manipulator. The concentric circles shown in the workspace are locations which can be reached in a singular configuration, the Jacobian surfaces. The annular regions bounded by the Jacobian surfaces are the w-sheets, $\mathcal{W}_i$, $i = 1, \cdots, 4$. In the configuration space Θ, region A is the inverse image of the w-sheet $\mathcal{W}_1$, $A = f^{-1}(\mathcal{W}_1)$. The disjoint regions B_1 and B_2 compose $f^{-1}(\mathcal{W}_2)$, $B_1 \bigcup B_2 = f^{-1}(\mathcal{W}_2)$. Region C is $f^{-1}(\mathcal{W}_3)$. The disjoint regions D_1 and D_2 are $f^{-1}(\mathcal{W}_4)$, $D_1 \bigcup D_2 = f^{-1}(\mathcal{W}_4)$. Because the singularities do not depend on joint angle θ_1 they form Jacobian surfaces of constant radius in $\mathcal{W}$, and thus the $\mathcal{W}_i$ form annuli.[2]

The preimages of the w-sheets, $f^{-1}(\mathcal{W}_i)$, are seen to consist of disjoint regions in Θ, denoted by $\mathcal{B}_i^j$, $j \in \{1, 2\}$, $\bigcup_j \mathcal{B}_i^j = f^{-1}(\mathcal{W}_i)$. Generically $x \in \mathcal{W}$ has as its inverse image $f^{-1}(x)$ either one or two self-motion manifolds in Θ. Each self-motion manifold is a connected set. The significance of the regions $\mathcal{B}_i^j$ is that for all $x \in \mathcal{W}_i$, exactly one self-motion manifold is in each $\mathcal{B}_i^j$. These $\mathcal{B}_i^j$, $\bigcup_j \mathcal{B}_i^j = f^{-1}(\mathcal{W}_i)$, are called solution branches in this paper.

Formally, a solution branch of $f(\cdot)$ over a w-sheet $\mathcal{W}_i$ is defined to be a maximally connected subset of $f^{-1}(\mathcal{W}_i)$. In Fig. 1, $f^{-1}(\mathcal{W}_1)$ and $f^{-1}(\mathcal{W}_3)$ each have a single solution branch, namely the regions $\mathcal{B}_1^1 = A$ and $\mathcal{B}_3^1 = C$. The preimages $f^{-1}(\mathcal{W}_2)$ and $f^{-1}(\mathcal{W}_4)$ each have two solution branches, regions $B_1 = \mathcal{B}_2^1$ and $B_2 = \mathcal{B}_2^2$ for $f^{-1}(\mathcal{W}_2)$ and regions $D_1 = \mathcal{B}_4^1$ and $D_2 = \mathcal{B}_4^2$ for $f^{-1}(\mathcal{W}_4)$. In this paper, it will be shown that the individual solution branches are identifiable by learning methods.

An important point to note is that, for a planar manipulator, the disjoint solution branches $\mathcal{B}_i^j$ have trivial fiber bundle topology [14]. A trivial fiber bundle is essentially a product space where one part of the product maps one-to-one onto the workspace, and the other part, the "fiber," is the self-motion manifold. In this case a canonical parameterization of fibers exists such that fixing the value of the parameter at any point in its range results in an inverse function $g(\cdot)$ over the workspace [21]. An inverse function is equivalent to a cross section of the fibers. Fig. 2 illustrates the fiber bundle notion for a single solution branch $\mathcal{B}_i^j$. The entire branch is called the total space, and is topologically equivalent to a product of a base space with the fiber $\mathcal{F}$. The function ϕ_{ij} separates the manifolds into the product of the canonical fiber, S^1, and a space isomorphic to the workspace. Its inverse ϕ_{ij}^{-1} serves as a parameterized inverse function; specifying a value for $s \in \mathcal{F} = S^1$ results in a unique configuration in Θ. Below we show the use of learning methods to approximate ϕ_{ij} over the invertible w-

[2] Changing θ_1 is equivalent to a simple rotation of the coordinate system of the workspace, which has no effect on the singularity structure of the manipulator.

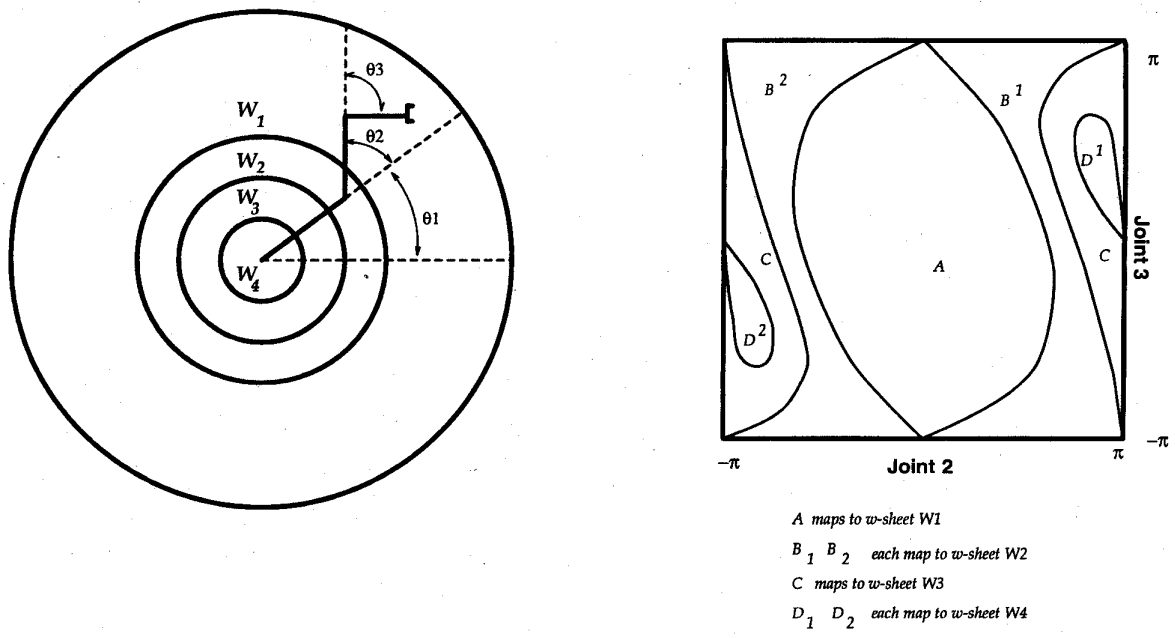


Fig. 1. The range space ("workspace") and the domain space ("configuration space" or "joint space") of the kinematic map for the 3R planar manipulator. The four annular regions are separated by boundaries (called Jacobian surfaces) which are the image of the singularities of the forward kinematic function. These regions are denoted by $\mathcal{W}_i, i = 1, \cdots, 4$. The configuration space is the three–torus, which is shown cut and "unfolded" into a cube, and projected to the Joint 2–Joint 3 plane.

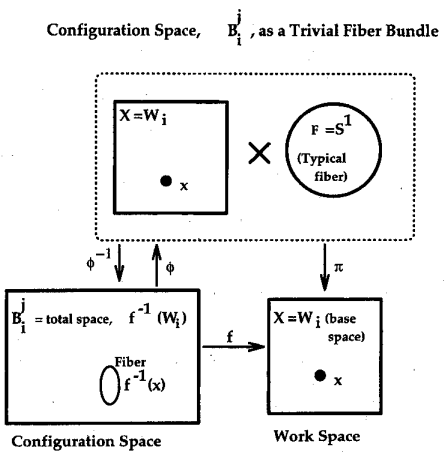


Fig. 2. The solution branch $\mathcal{B}_i^j$ as a trivial fiber bundle. The fiber over $x, f^{-1}(x)$, is the self-motion manifold. There exists a continuous, invertible function $\phi = \phi_{ij}$ such that the action of ϕ essentially "flattens out" $\mathcal{B}_i^j$ into a product space. ϕ_{ij} transforms the representation of a set of joint angles, θ, homeomorphically into the corresponding end-effector location, x, and a location l on the typical fiber, $\mathcal{F} = S^1$. Thus ϕ consists of f and an additional coordinate function $\mathcal{T}_{ij}$ which computes $\mathcal{T}_{ij} : \theta \to s \in F$. ϕ_{ij}^{-1} effectively inverts f over $\mathcal{W}_i$, parameterized by $s \in \mathcal{F}$. The function π denotes the canonical projection $\pi(x, s) \mapsto s$.

sheets. Fig. 3 details the fiber bundle structure of one of the w-sheets; the inverse image of any workspace location x is $f^{-1}(x)$, which is topologically equivalent to the circle S^1 for all x in the w-sheet.

The standard example of a fiber bundle which is not trivial is the Möbius band, an object which is locally the product of a circle and a line segment. The fiber is the unit interval I. The fiber bundle is locally $I \times S^1$ but with a half-twist in I going once around the circle. Any assignment of values in [0, 1] to each fiber with 0 at one end of the interval and 1 at the other is locally continuous, but not globally. Inverse functions can be constructed for nontrivial fiber bundles, but there must be a parameter value for which any given inverse function will be undefined. For a trivial fiber bundle, all parameter values result in a well-defined inverse function.

In the case of the 3R planar manipulator, the particular fiber is always diffeomorphic to the circle, S^1. S^1, however, can be embedded in the configuration space, T^3, in a number of ways. Two manifolds which cannot be smoothly transformed into one another[3] are said to be of different homotopy classes. The inverse image of a point in one w-sheet may be of a different homotopy class than that of a point in a different w-sheet. In general, the transition between w-sheets effects a bifurcation where either the number or homotopy class (or both) of the inverse solution changes. Fig. 4 illustrates the three homotopy classes for the specific manipulator considered.

Topological results developed in [14], [22], and [23] and described in detail in [17] guarantee the existence of a parsimonious disjoint partition of the 3R manipulator configuration space Θ into a finite set of trivial fiber bundles $\mathcal{B}_i^j$, each having as its base space the workspace w-sheet region $\mathcal{W}_i = f(\mathcal{B}_i^j)$, under the operation of the forward kinematics function restricted to the bundle, $f|_{\mathcal{B}_i^j}$. The w-sheets are the maximal workspace regions containing no critical values. The fiber bundles $\mathcal{B}_i^j$ are precisely the solution branches.

The solution to the inverse kinematics problem for the redundant 3R planar manipulator developed in this paper

[3] In a sense discussed more rigorously in [17].

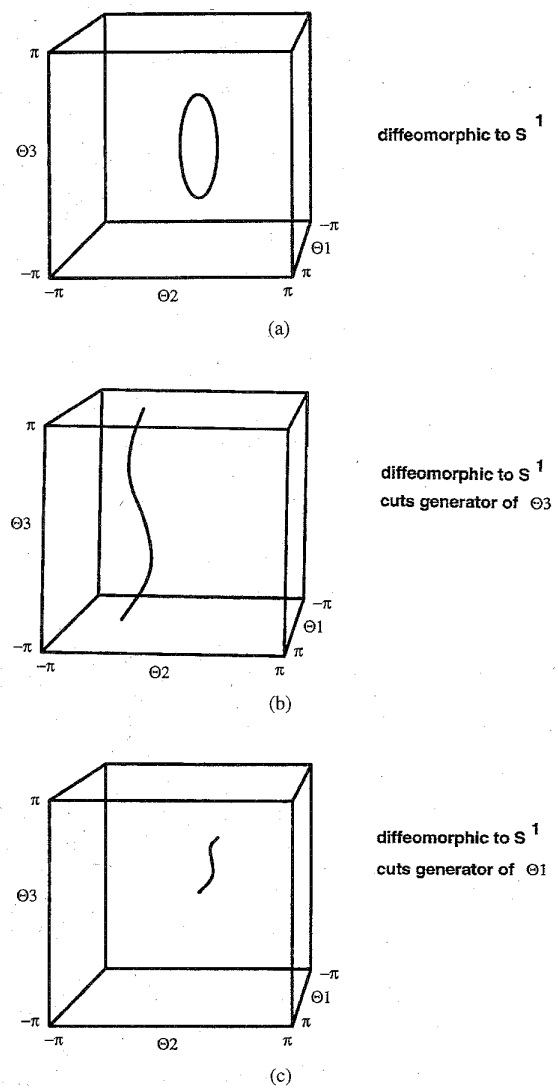


Fig. 3. Fibration of $A = f^{-1}(\mathcal{W}_1)$. Here, $A = \mathcal{B}_1 = f^{-1}(\mathcal{W}_1)$. All points along each of the one-dimensional fibers, $f^{-1}(x_i)$, map to the same point x_i in the range space. Each fiber represents the null-motion of the robot; the manipulator's end-effector remains at the same location in $\mathcal{W}_1$ for all configurations on the fiber. Each fiber is diffeomorphic to the "canonical fiber," S^1. The cross section shown for $s = s_0$ is $\phi_1^1(\mathcal{W}_1, s_0)$, which is diffeomorphic to $\mathcal{W}_1$.

exploits these topological facts to develop methods which process measured input–output data pairs (θ, x) for the following purposes:

1) Partitioning the workspace $\mathcal{W}$ into disjoint singularity-free w-sheet regions $\mathcal{W}_i$,
2) For each w-sheet $\mathcal{W}_i$, partitioning $f^{-1}(\mathcal{W}_i)$ into disjoint solution branch regions, $\mathcal{B}_i^j$,
3) For each solution branch region $\mathcal{B}_i^j$, exploiting the trivial fiber bundle structure of $\mathcal{B}_i^j$ to produce a canonical parameterization of the self-motion manifolds, and
4) Approximating parameterized smooth direct inverse functions from each w-sheet $\mathcal{W}_i$ to each of its solution branch regions.

Step 1) is briefly described below in Section IV. Step 2) is briefly discussed in Section IV and more fully in [17] and [18]. A significant contribution of this paper is the development of methods for achieving Step 3) by the use of a set of SOM's to approximate the self-motions. The SOM's are constructed so that the induced parameterization is smooth throughout a solution branch. Step 4) follows from Step 3) and is discussed in Section V.

III. Inverse Kinematics of Redundant Manipulators

In this section we discuss the inverse kinematics problem for redundant manipulators in general, and briefly review methods of resolving the redundancy.

It is desirable for a solution to the inverse kinematics problem to be both local and cyclic [24]. A local algorithm is one for which the solution can be determined from local path information; that is, knowing the current end-effector location and the target location suffice to find a set of joint angles which solve the inverse problem for the target. A cyclic algorithm is one which tracks a closed path in the workspace with a closed path in the configuration space. Cyclicity is important in industrial robotic applications because it is crucial in many applications that manipulator motions repeat in a regular and predictable manner. We show that constructing direct inverse functions achieves these goals.

A. Algorithmic Methods

Pseudoinverse Methods: Differentiating the kinematics function (1) above, results in

$$\dot{x} = J(\theta)\dot{\theta} \tag{3}$$

where $J = \partial f/\partial\theta$ is the manipulator Jacobian matrix. The pseudoinverse techniques specify a joint velocity $\dot{\theta}$ for a desired end-effector velocity $\dot{x}$ by finding the minimum norm solution to (3), $\dot{\theta} = J^+\dot{x}$, where J^+ denotes a pseudoinverse of J [25]. This achieves an indirect solution to the inverse kinematics problem when $\dot{x}$ is taken as the vector between the current position and the target position, $\dot{x} = x - x_d$, and one follows the trajectory in Θ defined by $\dot{\theta} = J^+\dot{x}$.

It is known that this approach will not always avoid singular configurations [26]. The pseudoinverse technique also does not have cyclicity; that is, following a closed trajectory in the workspace will not always correspond to a closed trajectory in the jointspace. The pseudoinverse techniques have many modifications, and a complete discussion is beyond the scope of this paper. A more detailed discussion can be found in [27].

Optimization Methods: Techniques based on global optimization methods have also been used [28]. Consider the manipulator's current position $x_0 = x(t_0) = f(\theta_0)$, and the target position $x_d = x(t_1)$. In this approach, the trajectory $\theta(t)$ is found which minimizes

$$\int_{t_0}^{t_1} \left(\frac{1}{2}\dot{\theta}^T W^{-1}\dot{\theta} + g(\theta)\right) dt$$

subject to the kinematics constraint that $x(t) = f(\theta(t))$. This indirectly solves the inverse kinematics problem for a target x_d, because $x_d = f(\theta(t_1))$. The functional g explicitly adds a side constraint, such as optimizing manipulability [29]. W is a user specified symmetric positive definite matrix weighting the

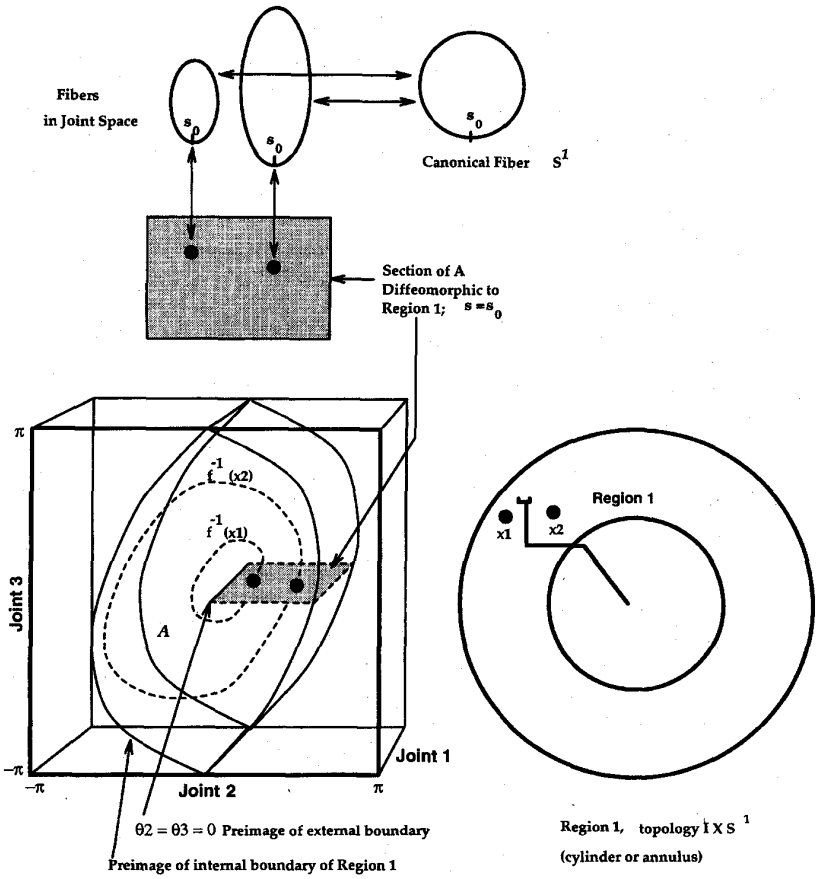


Fig. 4. Homotopy classes of self-motion manifolds in T^3 for the 3R planar manipulator. The three homotopy classes are shown. The configuration space is T^3, shown here "sliced" along a generator for each of θ_1, θ_2, and θ_3 and "unfolded" into a cube. In (a), a self-motion manifold, the fiber, which cuts no generators, is shown. In (b), the fiber wraps around T^3, cutting the generator for θ_3 (the ends connect at the top and bottom.) In (c), the fiber shown cuts the generator for θ_1. There is no way to smoothly deform any one of these into any other, thus they belong to separate homotopy classes.

terms. It has been shown that this method does not guarantee that singularities will be avoided [30], regardless of cost function g.

Lyapunov Methods: A third method (the Lyapunov or gradient descent method) based on Lyapunov stability theory has been developed [31], [32]. We follow [31] in the derivation.

Let θ be the current joint angle configuration. For a desired target end-effector location x_d, define positioning error as $\tilde{x} = f(\theta) - x_d = x - x_d$. A candidate Lyapunov function is the scalar error function $V = \frac{1}{2}\tilde{x}^T A \tilde{x}$ where A is positive definite. Differentiating

$$\dot{V} = \tilde{x}^T A \dot{x} = \tilde{x}^T A J(\theta)\dot{\theta}.$$

An indirect gradient based solution can be developed by updating θ by the rule

$$\dot{\theta} = -J(\theta)^T A^T \tilde{x}$$

by which

$$\dot{V} = -(J(\theta)^T A^T \tilde{x})^T (J(\theta)^T A^T \tilde{x}) = -\dot{\theta}^T \dot{\theta} \leq 0.$$

Since asymptotically $\dot{V} \to 0$ as $t \to \infty$, we have $\dot{\theta} \to 0$ [33]. Note that $\dot{\theta} = -J(\theta)^T A^T \tilde{x}$ is zero when either $\tilde{x} = 0$ (and thus the actual position is the desired position x_d) or when $A^T \tilde{x}$ is in the nullspace of $J(\theta)^T$. Thus if $J(\theta)$ is singular, a fixed point may be reached with nonzero positioning error $\tilde{x}$.

Augmentation Method: A fourth means of solving the inverse kinematics problem resolves the redundancy by introducing $n - m$ additional constraint functions, either directly in task space [47], or by adding $n - m$ rows to the nonsquare Jacobian, and then follow a trajectory defined by Lyapunov, optimization, or pseudoinverse methods on the resulting "extended Jacobian" [34]. This method is locally cyclic [13] and thus has advantages over the other methods listed above. Its main disadvantage is that algorithmic singularities can be introduced; the extended Jacobian may be singular at configurations where the end-effector Jacobian is still full-rank.

Note that the pseudoinverse, optimization, and Lyapunov methods are all computationally intensive, being recurrent, indirect (as defined in Section I) methods of solving the inverse kinematics problem.

Direct Inverse Kinematic Functions: Wampler defines an inverse kinematic function to be a continuous function g on a subset $\mathcal{W}_i$ of $\mathcal{W}$, g: $\mathcal{W}_i \to \Theta$, such that $f(g(x)) = x$, $\forall x \in$

$\mathcal{W}_i$ [12]. The function g maps a workspace location to a unique joint configuration.

There are any number of ways to build inverse functions. For example, recall that $f(\theta) = x$, where θ is a vector of n joint angles, $(\theta_1, \cdots, \theta_n)$. Choosing fixed, arbitrary values for $n-m$ of these, say θ_{m+1} to θ_n, results in a forward kinematic function of the remaining joint angles with no extra dof, which may be invertible in closed form. Inverse functions can also be developed by adding $n-m$ constraint equations to (1), as in the augmentation methods above.

Wampler defines an invertible workspace to be any subset of $\mathcal{W}$ on which there exists an inverse kinematic function [24]. For many workspaces, including the workspace of the 3R manipulator, it is known that no inverse function can exist globally over the entire workspace [12], [13]. There appears to be no general method for determining if a workspace is globally invertible, although it is often easy to determine when it isn't.

A cross section of a fiber bundle is defined by a continuous function $g : x \in X \rightarrow \theta \in B$ such that $f(g(x)) = x$ [20]. Any cross section through a trivial bundle will define an inverse function over the entire base space $X = \mathcal{W}_i \subset \mathcal{W}$ [20]. Further, if a workspace region $\mathcal{U} \subset \mathcal{W}$ contains no critical values, then $\forall x \in \mathcal{U}, f^{-1}(x)$ will consist of the same number of solution branches, and all will be diffeomorphic[4] to one another. Consequently we have Lemma 1.

Lemma 1: A subregion $\mathcal{U}$ of $\mathcal{W}$ is invertible if it contains no critical values and $f^{-1}(\mathcal{U})$ is a fiber bundle for any planar manipulator. In particular, all w-sheets $\mathcal{W}_i$ are invertible.

Inverse functions are local and cyclic. Because function evaluation is typically fast, inverse functions will in general have the additional property that they may be fast enough to be suitable for real-time control if an efficient computation of the approximate inverse is obtainable.

Because the w-sheets form the most parsimonious partition of the workspace into invertible regions, the set of inverse functions consisting of one parameterized family of inverse functions per solution branch per w-sheet will contain the minimum number of inverse functions necessary to cover the workspace such that all possible configurations are allowed for solutions. The families of inverse functions are parameterized by a natural representation of the redundant degrees of freedom defined by the canonical fiber.

B. Neural-Network Approaches

Several researchers have applied learning methods to the regularization of the underdetermined inverse kinematics problem. In general, a network is trained to produce as output the joint angle values corresponding to end-effector location input data generated from a redundant manipulator. To avoid the one-to-many problem, typically the values are constrained such that only one set of joint angles can be produced from any target position. Kim and Yoon [7] and Ahmad and Guez [5] regularize the three dof planar manipulator explicitly at training time, by generating training data and computing the joint angles which minimize the optimization criterion $M(\theta) = \sqrt{\det(J(\theta)J^T(\theta))}$ (the manipulability measure introduced by [29]). Similarly, Brüwer and Cruse [9] optimize over a cost function which penalizes extreme joint angles.

Jordan and Rumelhart [6] suggest first training a network to model the forward kinematics, $\theta \mapsto x$, then prepending another network. Training the cascaded networks, without further adaption of the weights of the forward model, to learn the identity, $x \mapsto x$, results in the prepended portion of the network learning an inverse. Specific optimization criteria can be included in the error in a principled way, resulting in an inverse achieving the desired optimization.

All of the above are direct methods which regularize the underconstrained (one-to-many) inverse kinematics problem by adding constraints at training time so that the problem becomes one-to-one. The regularization ensures that the network will find a solution, and can produce smooth trajectories, optimizing training-time criteria. At run-time, however, the extra dof cannot be exploited for other purposes nor can the optimization criteria be changed.

Such training time regularization may be unneccesary, depending on the inverse kinematics method used. For example, the indirect Lyapunov method discussed above can easily be implemented by a recurrent neural network [6], or developed within the framework of a bidirectional associative memory, [6], [11], [36]–[38]. Let $\hat{f}(\cdot)$ be a neural network approximating the forward kinematics function $f(\cdot)$. Let the current configuration, the input vector, be θ, the current end-effector location be approximated by $\hat{x} = \hat{f}(\theta)$, and the desired end-effector location be x_d. The positioning error is approximately $\tilde{x} = \hat{f}(\theta) - x_d = \hat{x} - x_d$. Let the network error be $\frac{1}{2}\tilde{x}^T A \tilde{x}$ for A positive definite. The backpropagation of this error to the input layer computes $\dot{\theta} = -\hat{J}(\theta)^T A^T \tilde{x}$ where $\hat{J} = \partial f / \partial \theta$. The Lyapunov method is thus approximated by iterative updates to θ, of the form $\Delta\theta = -\alpha \hat{J}(\theta)^T A^T \tilde{x}$. This computation effectively inverts the network by gradient descent.

The Lyapunov method has three potential drawbacks. First, as discussed in Section III-A, $\Delta\theta$ can be zero even when $\tilde{x}$ is large, if $J(\theta)$ has less than full rank. An heuristic solution to this problem is given by [36], where if $\tilde{x}$ is large when the update rule prescribes near-zero parameter changes, a "jumping" mechanism is invoked which moves the manipulator to another region in the configuration space. Second, the Lyapunov method does not result in cyclicity, even locally. Third, there may be multiple solution branches, in which case the method of [6] results in an inverse kinematics solution on the same branch as the initial configuration, while that of [36] results in a solution on a branch dependent on the jumping mechanism, and which is essentially random if jumping is invoked. Thus if the globally optimal solution is on another branch, it may not be found.

Ritter *et al.* [35] use SOM's to learn a specific cross section of the fiber bundle for one solution branch. This results in construction of an inverse function and achieves a high-accuracy solution. It resolves the redundancy at training time, however, to satisfy optimization criteria imposed ahead of time and computed off-line. Below we show that SOM's can be used to approximate the individual fibers, resulting in a map

[4] A diffeomorphism is a map which is one-to-one, onto, continuous, and differentiable, and with the inverse also differentiable.

of the entire configuration space fiber bundle rather than a single cross section.

C. Proposed Method: SOM's of Fibers

Unsupervised algorithms which produce SOM's from data have been developed and used by a number of researchers, see, e.g., [39]–[42]. Typically these algorithms operate to preserve neighborhoods on a network of nodes which encode the sample data. For example, Kohonen's algorithm is a vector quantization which preserves a neighborhood relation among the nodes, such that areas of high density in the data sample have higher density of nodes. Convergence results only exist for a small class of topologies of maps; however, the algorithms appear to work well in practice for maps of general topology [42].

The general form of an SOM consists of a set of simple processing elements (or nodes) which are topologically ordered, perhaps along several dimensions. Each node computes a nonlinear (generally scalar) function of points in the input space. Typically the function computed by each node is dependent on the distance between the input space point and a reference vector (or weight vector) associated with the node. The overall function computed by the map may be a linear or other combination of the individual nodes, or it may be the output of the single node whose reference vector is closest to the input space point.

The adaptation of a map to data consists of an iterative procedure for modifying the weight vectors, and node coefficients (if any). The data points are repeatedly "presented" to the network. For each data point, the processing element which has the largest output value is the "winner," and its weight vector is adjusted in the direction of that data point. The weight vectors of its neighbors, according to the topology of the net, are also adjusted, though typically by a lesser amount than the adjustment made to the winner.

Applications of SOM's to the robot inverse kinematics problem are given in [41], [2], and [4]. In the research reported in these papers, either the robot has no redundant dof, or the redundancy is resolved at training time, with only a single solution along a single branch available at run-time. Our results extend these prior approaches by providing a topology preserving mapping such that all solutions along all branches are accessible at run-time.

In Section IV we present a learning approach to constructing inverse kinematic functions by parameterizing the redundancy in a natural way with the use of SOM's. Using (θ, x) data generated from the forward kinematics function $f(\cdot)$, a parameterization of the redundancy is learned which corresponds to the fibers, $f^{-1}(x)$, being homeomorphic to the canonical fiber S^1. This parameterization can then be used to construct direct inverse kinematic functions over invertible subsets of $\mathcal{W}$ which are locally cyclic and have no algorithmic singularities.

The topological results given in Section II underlie the approach taken to learning direct inverse kinematic functions developed in this paper. The underconstrained inverse problem is regularizable over the w-sheets $\mathcal{W}_i$ (the open regions of the workspace bounded by the Jacobian surfaces) into a map to a fiber bundle consisting of the solution branches $\mathcal{B}_i^j$. Generating a canonical representation of the fibers then allows the construction of direct inverse functions parameterized by the redundant dof. To generate a canonical representation, the fibers over each w-sheet, $\mathcal{W}_i$, are parameterized with SOM's of the same topology.

The SOM, fit to the data, then yields a canonical parameterization. Values are assigned to the nodes (along each dimension if multidimensional) and locations between nodes are given an interpolated value. Networks of any topology can be fit to data from a distribution of any topology, however, the most useful parameterizations will come from using networks of the same topology as the data. In the 3R robotics case examined in this paper, the topology of the preimage manifolds is known (i.e., S^1 with one of three possible homotopy classes). Thus the network can be constructed to conform to the *a priori* known topology of the data and a parameterization of the data can be generated which follows the manifold.

IV. Learning Manipulator Self-Motions

The use of SOM's to obtain a learned parameterization of the self-motions of the 3R planar redundant manipulator is demonstrated in this section. The specific manipulator analyzed has link lengths of $l_1 = 0.4$ m, $l_2 = 0.3$ m, and $l_3 = 0.25$ m. The partitioning, fitting of SOM's, and construction of inverse functions used 216 000 input–output data samples (θ, x), where $\theta \in \Theta$ and $x \in \mathcal{W} = f(\theta)$. For the planar manipulator, the forward kinematics is

$$\begin{pmatrix} x \\ y \end{pmatrix} = f(\theta) = f(\theta_1, \theta_2, \theta_3) = \begin{pmatrix} l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \\ l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \end{pmatrix}.$$

The training data are generated by sampling on a regular grid in the configuration space Θ (every six degrees along each of the three joint angle dimensions) and computing the end-effector location $x = f(\theta)$ from the forward kinematics function. Note that actual measured data generated from a physical manipulator could also be used, in which case an explicit functional form of $f(\cdot)$ is not needed.

The precise topological structure of the configuration space depends on the specific link lengths. If the link lengths are such that the sum of any two are longer than the third, then the manipulator can reach its base, and the workspace is a disk of radius $l_1 + l_2 + l_3$ centered at the base. This is the case we are considering; the other cases have comparable topological structures [17].

A. Partitioning the Workspace and Configuration Space

The goal of this section is to demonstrate that the w-sheets can be identified by learning methods, making use of the topological knowledge of the nature of the w-sheets to identify their boundaries. We know that the w-sheets are invertible, thus once we can identify the individual w-sheets, we can then construct explicit inverse functions.

Because the singularity surfaces of an all-revolute manipulator are constant with respect to changes in the first joint

angle, the disjoint w-sheets $\mathcal{W}_i$ form annuli, as seen in Fig. 1. Thus the approximate location of the Jacobian surfaces (the separating surfaces in the workspace, $\mathcal{W}$) can be identified by a one-dimensional search along a radius from the center of $\mathcal{W}$ out to the workspace exterior boundary. The size of the steps are taken to be about one-third the distance of the shortest link of the manipulator (8 cm). Laying the first point 4 cm from the center, 12 equally spaced query points are generated along the radius. The data samples which have their x component lying within 4 cm of the each query point are retrieved.

The preimage data are then processed by a clustering algorithm which determines the number of preimage manifolds. As discussed in Section II, for the 3R manipulator, the number of self-motion manifolds changes between neighboring w-sheets, $\mathcal{W}_i$, the bifurcation occurring on the separating singularity surface [14]. If two neighboring query points differ in the number of identified self-motion manifolds, the location of the singularity surface must be between them, and here is assumed to be at their midpoint.[5] If a more accurate determination of the singularity surfaces is desired, finer grained sampling between the identified bounding query points can be made.

This process identifies four workspace regions, which approximate the w-sheets shown in Fig. 1. The outermost identified region is labeled $\mathcal{W}_1$, extending from the workspace boundary inward to a radius of 0.48 m (the actual singularity is at 0.45 m). $\mathcal{W}_2$ is identified as extending from 0.48 m to 0.32 m (the actual location of the boundary of this region is at 0.35 m). $\mathcal{W}_3$ is determined to extend from 0.32 m to 0.16 m (the actual innermost separating singularity is at 0.15 m). The innermost disk with radius less than 0.16 m is identified as $\mathcal{W}_4$.

Each identified w-sheet $\mathcal{W}_i$ is now handled separately. The points in $\mathcal{W}_2$ and $\mathcal{W}_4$ have two disjoint preimage manifolds, which lie in the solution branches $\mathcal{B}_i^j$, $i = 2, 4$, $j = 1, 2$. The clustering and merging procedure of [45] and [18] is used for solution branch identification; the preimage data for the points in $\mathcal{W}_2$ are partitioned into $\mathcal{B}_2^1$ and $\mathcal{B}_2^2$, and those for points in $\mathcal{W}_4$ are partitioned into $\mathcal{B}_4^1$ and $\mathcal{B}_4^2$.

A regular two-dimensional grid with 12 cm spacing is laid down over the set of data points in the workspace. Inside a box 6 cm on a side centered at the intersections of the grid lines, a "query point" is chosen randomly (uniformly). All data samples with x component within 3 cm of each of the query points are retrieved. Those query points and their associated data such that no less than 50 points lie in the 3 cm neighborhood are retained, the others are not used. This process produced 213 query points in $\mathcal{W}$. These are separated based on their distance from the origin of the workspace into the four $\mathcal{W}i$. $\mathcal{W}_4$ contains five query points, $\mathcal{W}_3$ contains 16, $\mathcal{W}_2$ contains 22 and $\mathcal{W}_1$ contains the remaining 170.

B. Parameterization with SOM's

We have now identified input–output data samples from each isolated branch $\mathcal{B}_i^j$ of a 3R planar manipulator ($\mathcal{B}_i^j$ forming a trivial fiber bundle of known topological structure, as discussed in Section II). The next step is to learn a canonical parameterization which is consistent throughout an entire fiber bundle $\mathcal{B}_i^j$ region. We do so by fitting SOM's to several reference fibers in the region and interpolating. Data along the reference fibers are selected from the database, as joint angle vectors which map to a workspace location near one of a set of query points.

Let the SOM for a particular query point x_q be denoted by SOM_q, and each of its nodes be denoted by $w_{q,k}, k = 1, \cdots, n_i$ (here, n_i, the number of nodes in SOM i, is 20). If the context is clear, the query point subscript "q" for a single node may be omitted. The SOM's are explicitly constructed for fibers representing the inverse data from a set of query points in each $\mathcal{W}_i$ region. Canonical "zero" reference points of neighboring fibers are kept close to each other.

As described in Section II, each solution branch $\mathcal{B}_i^j$ of the 3R planar positioner is known to be a fiber bundle with fiber diffeomorphic to the typical fiber S^1, and belonging to one of the three possible homotopy classes shown in Fig. 4. Thus, the topology of the fibers for the case being considered is a one-dimensional manifold diffeomorphic to the circle, S^1, belonging to one of the three possible homotopy classes. Therefore a topologically consistent SOM consists of a circular set of nodes. One such formulation is the "elastic network" of Durbin and Willshaw [43], [44]. The elastic network algorithm applied to data samples approximating one of the inverse fibers (corresponding to the self-motion for an end-effector location x_q) will produce a closed chain of nodes approximating the data. If the number of data samples is significantly greater than the number of nodes in the elastic network, the resulting network will smoothly approximate the data, as long as the data in fact lie along a one-dimensional closed loop in the joint angle space.

A set of nodes (a "ring") is used to parameterize each fiber using the Durbin–Willshaw algorithm

$$\Delta w_i = \eta\left(\sum_\mu \Lambda_\mu(i)(d_\mu - w_i) + \kappa(w_{i+1} + w_{i-1} - 2w_i)\right)$$

where d_μ is the location of the μth data point, w_i is the location of the ith node in the network, and $\Lambda_\mu(i)$ is a neighborhood function. Here

$$\Lambda_\mu(i) = \frac{e^{|d_\mu - w_i|^2/2\sigma^2}}{\sum_j e^{|d_\mu - w_j|^2/2\sigma^2}}.$$

Our terminology and notation are similar to that in [44].

A set of query points in the w-sheet $\mathcal{W}_i$ are selected and the corresponding joint space data for a single solution branch $\mathcal{B}_i^j$ is retrieved from the database of joint angle and position data. This joint space data approximates the fibers in $\mathcal{B}_i^j$. To initialize the process, an elastic network is fit to the data from the fiber over query point x_1. This trained SOM is next used as the initial SOM for fitting to the data for same solution branch of the fiber over the neighboring query point x_2, and similarly the SOM trained to the data for query point x_{q-1} is used as the initial SOM for fitting the data for the fiber over query point x_q.

[5] The accuracy with which the singularities can be determined from data only is dependent on the density of the data sampling in the region of the singularity, and on the granularity of the step sizes in $\mathcal{W}$.

For the fiber over x_q, the node $w_{q,0}$ is designated the canonical zero reference point of SOM_q. To coerce $w_{q,0}$ to be near $w_{q-1,0}$, we add a term to the learning algorithm for $w_{q,0}$, such that for adjusting $w_{q,0}$

$$\Delta w_0 = \eta\left(\sum_\mu \Lambda_\mu(0)(d_\mu - w_0) + \kappa(w_1 + w_s - 2w_0)\right) + \alpha(w_{q-1,0} - w_{q,0}) \tag{4}$$

where $w_{q-1,0}$ is the zero position of the initial network and α is an adaptation rate parameter, and u is the index of the last node in the SOM. The adaptation rate parameters, η, κ, and σ, are typically initialized to the same value (here, 0.4) and annealed in proportion to the number of epochs of data presentation.[6] The parameter α is initialized to a slightly higher value and annealed exponentially, such that it is the dominant term for the first 100 or so epochs, and afterwards is dominated by the other terms. This forces $w_{q,0}$ to be near $w_{q-1,0}$ initially, and later adjusting toward nearby data when the rest of the nodes in the network have converged. As a result of the training, we have, for each solution branch region $\mathcal{B}_i^j$, a set of nodes $w_{q,k},\ q = 1, \cdots, n_i,\ k = 0, \cdots, u$ lying on the fiber $f^{-1}(x_q)$. Let $\theta_{q,k}$ denote the joint angles in $\mathcal{B}_i^j$ at which node $w_{q,k}$ is located with corresponding query point, $x_q = f(\theta_{q,k})$ after training.

For each ring SOM_q the distances between nodes $w_k, k = 0, \cdots, u$, are computed (using the "flat" metric on the torus[7]) and the total distance around the ring found. Each node is given a parameter value equal to its normalized distance around the ring (from the canonical zero point). Points lying directly between nodes are given an interpolated parameter value. As a consequence of this procedure, parameter values are effectively given by a learned coordinate function $\mathcal{T}_{ij}$: $\theta \in \mathcal{B}_i^j \mapsto s \in S^1$, defined by the nodes $w_{q,n}$, which explicitly encodes a consistent value s for the redundant degree of freedom within a solution branch $\mathcal{B}_i^j$.

For each branch, a set of SOM's are trained to parameterize the preimage self-motion manifolds corresponding to the query points such that the parameterization is smooth between manifolds and matched to the appropriate homotopy class. The query points in the identified w-sheets $\mathcal{W}_i$ are ordered such that the distance from one to the next is small (essentially, a continuous path through $\mathcal{W}_i$ is constructed passing through the query points). The query points are identified as $x_q, q = 1, \cdots, n_i$, where n_1 is 170, n_2 is 22, n_3 is 16, and n_4 is 5, as above.

The procedure for parameterizing the solution branch region $\mathcal{B}_1^1 = f^{-1}(\mathcal{W}_1)$ is now described fully as an example. The θ components of the data samples lying in the neighborhood of query point $x_1 \in \mathcal{W}_1$ are retrieved. An SOM consisting of $u = 20$ nodes arranged in a topology of the appropriate homotopy class in Θ [namely that of Fig. 4(a)] is then fit to the θ points by the Durbin–Willshaw algorithm, annealing the parameters η, σ, κ inversely with the number of adaptation epochs.[8] Call this network SOM_1. Fig. 5 shows this network after one, five, 10, and 500 training epochs.

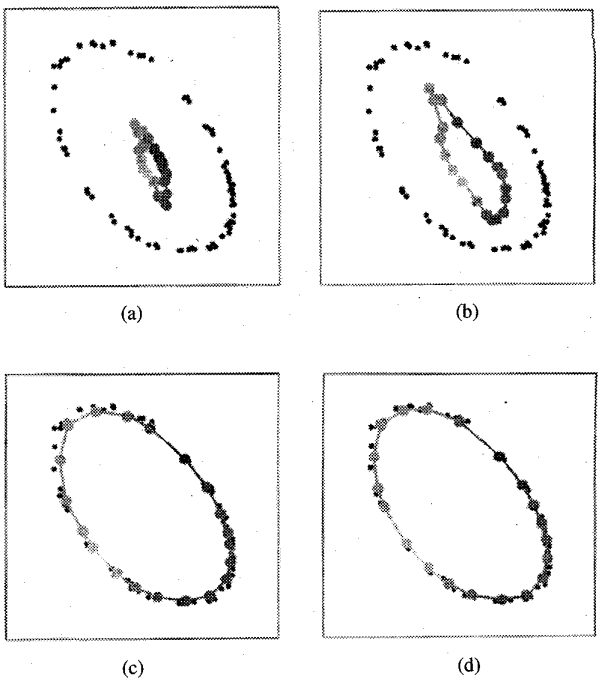


Fig. 5. Elastic network ($f^{-1}(x_1)$ in $\mathcal{B}_1^1$). The adaptation of the initial network in region $\mathcal{B}_1^1$ (a) after one epoch, (b) after five epochs, (c) after 10 epochs, (d) after training is completed, 500 epochs. The data points are shown projected to the θ_2–θ_3 axis; the axes range from $-\pi$ to π and opposite edges are identified (wrap around).

The θ points in the preimage of the neighborhood of x_2 are next retrieved, and SOM_1 is used as the initial value for SOM_2. One of the nodes of SOM_1 is arbitrarily chosen to be the zero point (labeled $w_{1,0}$), and the corresponding node in SOM_2, $w_{2,0}$, is updated during the adaptation to this set of data as given in (4), above, forcing it to be close to $w_{1,0}$ as well as adjusting toward the data in its region of influence.[9]

SOM's are fit to the inverse data for all of the query points in each $\mathcal{W}_i$, for each $\mathcal{B}_i^j$. This results in a collection of SOM's for each region. Each SOM approximates the fiber for its query point. In Fig. 6, the reference fibers approximating the fibration in the configuration space over the annulus $\mathcal{W}_3$ in Fig. 1 is shown. Twenty locations in $\mathcal{W}_3$ are selected and networks fit to the inverse data corresponding to a set of points (approximating a fiber) which map to one of these query points.

The parameterization is roughly equivalent to an angle, since the fiber is diffeomorphic to the circle S^1, thus they can be assigned a value s in $[0, 2\pi)$. If a smooth representation is desired, these values can be converted to $(\cos s, \sin s)$ pairs, eliminating the discontinuity at zero (at the cost of a two parameter representation of S^1). These values have been

[6] One "epoch" is the presentation of the entire set of data, in this case, the θ for the query point. The η, κ, and σ are reduced in proportion to the number of epochs n, while α is set equal to 0.95^n.

[7] Let two points on the n torus be $x = (x_1, \cdots, x_n)$ and $y = (y_1, \cdots, y_n)$. We use the metric $D(x, y) = \sum_{i=1}^n \|x_i - y_i\|^2$ where $\|x_i - y_i\| = \min(|x_i - y_i|, 2\pi - (|x_i - y_i|))$.

[8] Where one epoch consists of an adaptation step for each of the data points. Let $\eta_0, \sigma_0, \kappa_0$ be the initial values, and let $\eta_n, \sigma_n, \kappa_n$ be the values at the nth training epoch. $\eta_n = \eta_0/\sqrt{n}, \sigma_n = \sigma_0/\sqrt{n}$, and $\kappa_n = \kappa_0/\sqrt{n}$.

[9] This term is annealed exponentially, $\alpha_n = \alpha_0(0.95)^n$.

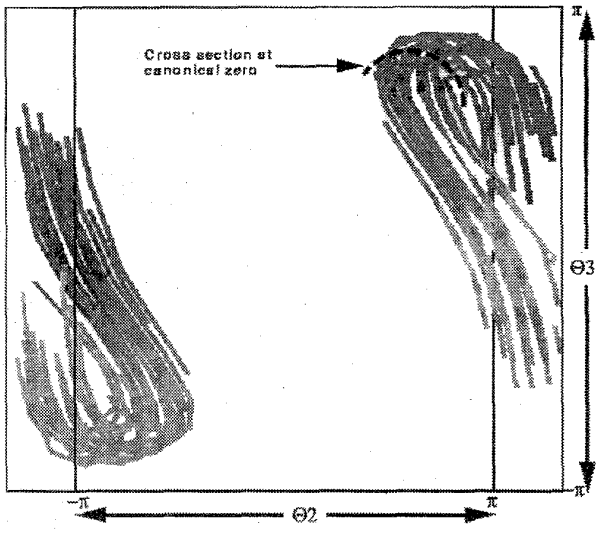


Fig. 6. Fibration $\mathcal{B}_3^1$. Twenty fibers in the inverse image of $\mathcal{W}_3$. These parameterize region C of Fig. 2. Apparent discontinuities are due to the representation of the three-torus as a cube—opposite faces of the cube are identified with each other. The pixel intensity represents an induced parameter value based on normalized distance froma canonical zero point. The range along each axis is $-\pi$ to π.

mapped to grayscale intensity in Fig. 6. It can be seen that the parameterization is qualitatively consistent over the set of fibers.

For any point in the branch $\mathcal{B}_i^j$, the nodes in all of the elastic nets can be used to approximate ϕ_{ij} and ϕ_{ij}^{-1}. For example, one can approximate ϕ_{ij} with a feedforward network using all of the nodes on all of the SOM's and their parameter values as training examples. Typically, the joint angles are represented as (cos, sin) pairs with the output nodes representing x, y, $\cos s$, and $\sin s$. The outputs are postprocessed to ensure that $\cos^2 + \sin^2 = 1$. The parameter s can be extracted as $\mathrm{Atan}\,2(\cos s, \sin s)$. The approximation of ϕ_{ij}^{-1} is discussed in more detail in the next section. Note that ϕ_{ij} is composed of two parts, the forward kinematics function computing $f(\theta)$ over $\mathcal{B}_i^j$, and a self-motion coordinate function $\mathcal{T}_{ij}$ mapping each θ to a location on the typical fiber $F = S^1$. See Fig. 2.

V. Regularized Inverse Kinematic Functions

The trained set of SOM's can now be used to compute a parameterized direct inverse kinematics function. The learned self-motion coordinate function $\mathcal{T}_{ij}$ is used to augment the sampled data with a value on S^1. The augmented mapping $\phi_{ij} = (f(\cdot)|_{\mathcal{B}_i^j}, \mathcal{T}_{ij}(\cdot))$: $\theta \in \mathcal{B}_i^j \mapsto (x, s)$ (where $x \in \mathcal{W}_i$ and $s \in S^1$) is one-to-one and onto, and thus invertible. The SOM's can be used to compute joint angles from a given target end-effector and solution branch location (x_d, s_d), effectively inverting ϕ_{ij} as we will now show.

The x and y components of x for our example 3R manipulator range from $-.95$ m to $+.95$ m. Points on S^1 are represented as a pair, $(\cos s, \sin s)$, for s in the range $[0, 2\pi)$. A locally Euclidean metric is used to compute distances between two workspace points

$$d((x_1, s_1), (x_2, s_2)) = ((x_1 - x_2)^2 + (y_1 - y_2)^2 + (\cos s_1 - \cos s_2)^2 + (\sin s_1 - \sin s_2)^2)^{1/2}$$

where x_i and y_i are the x and y components of x_i. Alternative metrics can be chosen which trade off physical distance in $\mathcal{W}$ for parameter distance in S^1. Letting $\hat{x}$ represent a point in $\mathcal{W} \times S^1$, $\hat{x} \triangleq (x, \cos s, \sin s)$, define $d(\hat{x}_1, \hat{x}_2) \triangleq d((x_1, s_1), (x_2, s_2))$, and $\phi_{ij}(\hat{x}) \triangleq \phi_{ij}(x, s)$.

In each solution branch region $\mathcal{B}_i^j$, we use a 20-node SOM for each query point x_q in $\mathcal{W}_i$. Each node $w_{q,k}$ associates a vector of joint angles with a point in $\mathcal{W} \times S^1$. Joint angles in $\mathcal{B}_i^j$ for the target $\hat{x}_d \in \mathcal{W}_i$ are computed as

$$\text{Branch } \mathcal{B}_i^j\text{: } \hat{\theta}(\hat{x}_d) = \frac{1}{\sum_{q=1}^{n_i} \sum_{k=0}^{u} e^{-d^2(x_{q,k}, \hat{x}_d)/\sigma^2}} \times \sum_{q=1}^{n_i} \sum_{k=0}^{u} \theta_{q,k} e^{-d^2(x_{q,k}, \hat{x}_d)/\sigma^2} \approx \phi_{ij}^{-1}(x_d, s_d) \tag{5}$$

where σ is an adjustable "receptive field" width. Note that the sum is over all SOM_q in $\mathcal{B}_i^j$, $q = 1, \cdots, n_i$. The weighting term ensures that a convex combination of all of the SOM nodes in $\mathcal{B}_i^j$ are used, with those closest to $\hat{x}_d$ weighted more heavily. A value for σ of 0.2 is used; the results are qualitatively the same for σ anywhere between 0.1 and 0.5. This is essentially a radial basis function network, with each node of each SOM being a unit with centroid given by its $\hat{x}$ coordinates in $R^2 \times S^1$, coefficients given by the associated θ values, and receptive field σ [35].

The inverse kinematics problem has now been solved by a parameterized family of direct inverse functions, where the parameterization is a learned one over the redundancy. Taking any continuous function $l(x)$: $x \mapsto s$ then results in an inverse function $g(x) = \phi_{ij}^{-1}(x, l(x))$ (restricted to the appropriate $\mathcal{W}_i$). In addition, a joint space trajectory along the self-motion can be computed by maintaining constant target end-effector position and continuously changing the redundancy parameter.

The positioning error of our example manipulator, when using (5) to locate the end-effector based on the SOM's determined in Section IV, averages 3.6 cm, which is about 2% of the diameter of the workspace, with a maximum error of 16.7 cm. The performance in $\mathcal{W}_1$ is the best, which is to be expected, given that it contains $170 \times 20 = 3400$ SOM nodes, but there is only small variation in positioning error across the regions. Table I shows the average positioning error in centimeters for targets in each region. The averages were computed from a random (uniform) sample of 10 000 points in the workspace. $\mathcal{W}_1$ contains 7774 of the total test points, $\mathcal{W}_2$ 830, $\mathcal{W}_3$ 1181 and $\mathcal{W}_4$ 215.

Ritter *et al.* [35] use an alternative approach to computing the joint angles. They maintain an approximation of the tangent space of the kinematics mapping defined piecewise at each SOM node. The node with its weight (denoted by w_s) closest to the x_d, the target end-effector position, is determined, and the joint angles for x_d are produced by

TABLE I

POSITIONING ERROR. THE AVERAGE POSITIONING ERROR PER WORKSPACE REGION IS GIVEN IN ABSOLUTE DISTANCE AND AS A PERCENTAGE OF THE 1.9M DIAMETER WORKSPACE. THE TRAINED SOM'S ARE USED TO APPROXIMATE THE REGULARIZED DIRECT INVERSE FUNCTIONS. THE RESULTING JOINT ANGLES ARE USED BY THE TRUE FORWARD KINEMATICS FUNCTION TO OBTAIN THE ACTUAL POSITION IN THE WORKSPACE, AND THE DISTANCE FROM THE TARGET COMPUTED. THE AVERAGES ARE COMPUTED OVER 10 000 WORKSPACE POINTS CHOSEN AT RANDOM, EACH WITH A RANDOM SETTING OF THE REDUNDANCY PARAMETER

Region	Error (cm_)	% Error
$\mathcal{W}_1$	3.20	1.68
$\mathcal{W}_2$	3.91	2.06
$\mathcal{W}_3$	5.33	2.81
$\mathcal{W}_4$	5.50	2.89
Total	3.57	1.88

an approximation of the first-order Taylor series about w_s. Essentially, their approach regularizes at training time by constructing a map for a single, specific cross section of the fiber bundle satisfying an optimization criterion assumed *a priori*. Using the parameterization developed by the methods in this paper, it should be possible to extend the work of [35] such that the entire self-motion is available at run-time.

As an alternative to (5), other nonlinear function approximators can be used; for example, augmented data pairs $(\theta,(x,s)) \in \mathcal{B}_i^j \times (\mathcal{W}_i \times S^1)$ can be found for each (θ, x) pair in $\mathcal{B}_i^j \times \mathcal{W}_i$, and can then be used to train a feedforward neural network to approximate ϕ_{ij}^{-1}: $(x,s) \mapsto \theta$.

Note also that the Lyapunov method discussed in Section III, and specifically the neural-network implementation of [6], can be enhanced by means of the augmented data, $(\theta,(x,s))$, generated by $\mathcal{T}_{ij}$. A forward model can be trained to approximate the augmented mapping $\theta \mapsto (x,s) = \hat{x}$. After instantiating the model with the current configuration, an error can be computed between the target $\hat{x}$ and the actual $\hat{x}$, and backpropagated to the inputs as per Section III-B above.

VI. DISCUSSION AND CONCLUSION

This paper has developed a means of parameterizing the trivial fiber bundles induced by the operation of the forward kinematics map on the configuration space of the three-link planar manipulator. We showed that topological knowledge can be used to regularize the ill-posed inverse kinematics problem for the redundant manipulator by partitioning the problem into a finite set of well-posed inverse problems over the solution branches $\mathcal{B}_i^j$. The redundancy is resolved by introduction of learned free parameters in a canonical way, exploiting the trivial fiber bundle structure of each solution branch. Regularized direct inverse functions are constructed which cover all of the reachable workspace for the three-link redundant planar arm using SOM's. In this manner we have extended the direct methods previously used only for nonredundant manipulators to the redundant manipulator case.

In the course of the work reported in this paper, a number of research questions have arisen for further investigation. These include

1) Automatic determination of the homotopy class of the fibers for various workspace regions.
2) Appropriate means of computing inverse kinematics smoothly for trajectories which span multiple w-sheets.
3) Use of the redundancy parameter s for flexible optimization of side criteria at run-time.
4) Cyclicity of trajectories which cross w-sheet boundaries.
5) Extension to the 4R manipulator for positioning in R^3.
6) Extension to systems with more than one redundant dof.
7) Accuracy of positioning/tracking.

Expanding on these points individually

1) The structure of the canonical fiber F is known from the geometry of the manipulator, namely $F = S^1$. Because the configuration space itself is not Cartesian, however, there are a variety of ways to imbed the fiber in the space as shown in Fig. 4. Two S^1 manifolds which cannot be smoothly transformed into one another are of different homotopy classes and there are three possible homotopy classes for the fibers in the case of the 3R planar manipulator. It should be easy to implement an automatic assessment by fitting a SOM of each homotopy class to data from a fiber in the region, and choosing the one which has the best fit.
2) Inverse functions are developed for each w-sheet $\mathcal{W}_i$ separately; when tracking a point in $\mathcal{W}$ across the w-sheet boundaries, there is no guarantee that s will vary smoothly, and indeed, in general it will not; a discontinuity may exist. Thus, for a boundary point x^* between $\mathcal{W}_i$ and $\mathcal{W}_{i+1}$, it may not always be possible to have $\lim_{\theta \in \Theta_i \to \theta^*} \mathcal{T}_i(\theta) = \lim_{\theta \in \Theta_{i+1} \to \theta^*} \mathcal{T}_{i+1}(\theta)$. When approaching the boundary, however, it is possible to determine which $\mathcal{W}_i$ the target is in, and compute an appropriate value of the redundancy parameter when a transition between $\mathcal{W}_i$ occurs. For example, let the trajectory include target $x_i \in \mathcal{W}_1$, with redundancy parameter value s_i, followed by $x_{i+1} \in \mathcal{W}_2$. $\theta_i = \phi_{ij}^{-1}(x_i, s_i)$. The fact is that there is a transition to $\mathcal{W}_2$ is detected, and both SOM's for x_{i+1} can be searched for the configuration θ_{i+1} closest to θ_i. The redundancy parameter can be set to $\mathcal{T}(\theta_{i+1})$, leading to a smooth trajectory in the configuration space and workspace. It remains to be seen whether it is possible to construct a smooth transition map between redundancy parameters for neighboring $\mathcal{W}_i$; it may be necessary for s to be discontinuous.
3) The excess dof can be used to find the inverse which both positions the end-effector at the desired location and optimizes a side constraint such as the manipulability index, $M(\theta) = \sqrt{\det(J(\theta)J^T(\theta))}$. In the case of the 3R planar manipulator, the maximum manipulability corresponds approximately to a constant value of the redundancy parameter over a solution branch $\mathcal{B}_i^j$ [17]. For other objective functions the optimal configuration is located on the one-dimensional self-motion manifold, and thus can be found by a one-dimensional search, rather than a search through the entire three-dimensional configuration space. Consequently, formulating inverse functions as in this paper should lead to improved means of identifying globally optimal configurations for criteria which may be imposed at run-time.
4) The inverse functions which are constructed are deterministic, smooth functions, and thus are guaranteed

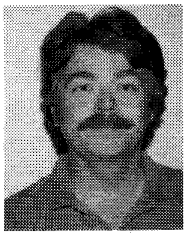
to be locally cyclic within a single w-sheet $\mathcal{W}_i$. If the method of mapping redundancy parameters when transitioning between $\mathcal{W}_i$ is invertible, then tracking will be globally cyclic, provided that kinematically singular configurations are avoided.

5) The extension to a 4R regional manipulator is expected to be straightforward. The method should generalize to redundant regional (positioning in R^3) manipulators, and indeed, to general smooth mappings over compact manifolds. The difficulties will arise in identifying the submanifolds which are trivial fiber bundles. In the case of the regional 4R manipulator, the w-sheets $\mathcal{W}_i$ will only have a finite number of holes. Thus they can be partitioned into a small, finite number of subsets for which the preimage is trivial, as discussed in Section II. There may be difficulty "pasting" together the direct inverse functions at the boundaries.
6) More challenging will be systems with two or more excess dof. The topological structure of the fibers of a 4R planar or 5R regional manipulator are $S^1 \times S^1 = T^2$, which can be embedded in T^4 or T^5 in many ways. The ideas of this paper should be applicable, although the implementation may be complex.
7) Because the methods used herein approximate the direct inverse functions from measured data, accuracy will depend on the sampling density and measurement error. It is unlikely that accuracy suitable for industrial applications can be obtained simply from the inverse functions. The inverse solution, however, is obtained extremely fast, and is nearby the zero-error solution in configuration space. Using the solution from the direct inverse function as a starting point, any of the differential methods can be employed to quickly iterate to a low error solution, assuming that position or velocity error can be measured in real time.

REFERENCES

[1] V. Guillemin and A. Pollack, *Differential Topology*. Englewood Cliffs, NJ: Prentice-Hall, 1974.
[2] T. M. Martinetz, H. J. Ritter, and K. J. Schulten, "Three-dimensional neural net for learning visuomotor coordination of a robot arm," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 131–136, 1990.
[3] S. Lee and G. Bekey, "Applications of neural networks to robotics," in *Control and Dynamic Systems, Advances in Theory and Applications, vol. 39: Advances in Robotic Systems, Part I*, C. T. Leondes, Ed. New York: Academic, 1991, pp. 1–69.
[4] J. A. Walter and K. J. Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot," *IEEE Trans. Neural Networks*, vol. 4, no. 1, pp. 86–94, 1993.
[5] Z. Ahmad and A. Guez, "On the solution to the inverse kinematics problem," in *Proc. IEEE Int. Conf. Robotics Automat.*, Cincinnati, 1990, pp. 1692–1697.
[6] M. I. Jordan and D. E. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive Sci.*, vol. 16, pp. 307–354, 1992.
[7] K.-Y. Kim and Y.-S. Yoon, "Practical inverse kinematics of a kinematically redundant robot using a neural network," *Advanced Robotics*, vol. 6, no. 4, pp. 431–440, 1992.
[8] R. Eckmiller, J. Beckmann, H. Werntges, and M. Lades, "Neural kinematics net for a redundant robot arm," in *Proc. IJCNN*, vol. II, Washington, D.C., 1989, pp. 333–340.
[9] M. Brüwer and H. Cruse, "A network model for the control of the movement of a redundant manipulator," *Biol. Cybern.*, vol. 62, pp. 549–555, 1990.
[10] F. Pourboghrat, "Neural networks for learning inverse kinematics of redundant manipulators," in *Proc. 32nd Midwest Symp. Circuits Sys.*, 1990, pp. 760–762.
[11] S. Lee and R. M. Kil, "Robot kinematic control based on bidirectional mapping network," in *Proc. 1990 IJCNN*, San Diego.
[12] C. W. Wampler II, "The inverse function approach to kinematic control of redundant manipulators," in *Proc. Amer. Contr. Conf.*, Atlanta, 1988.
[13] D. R. Baker and C. W. Wampler, "On the inverse kinematics of redundant manipulators," *Int. J. Robotics Res.*, vol. 7, no. 2, pp. 3–21, 1988.
[14] J. Burdick, "Kinematics and design of redundant robot manipulators," Ph.D. dissertation, Dep. Mech. Eng., Stanford Univ., 1988.
[15] D. DeMers and K. Kreutz-Delgado, "Issues in learning global properties of the robot kinematics mapping," in *Proc. IEEE Int. Conf. Robotics Automat.*, Atlanta, 1993, pp. 205–212.
[16] D. Kohli and M.-S. Hsu, "The Jacobian analysis of workspaces of mechanical manipulators," *Mech. Mach. Theory*, vol. 22, no. 3, pp. 265–275, 1987.
[17] D. DeMers, "Learning to invert many-to-one mappings," Ph.D. dissertation, Dep. Comput. Sci. Eng., Univ. California, San Diego, 1993.
[18] D. DeMers and K. Kreutz-Delgado, "Learning global properties of nonredundant kinematic mappings," Inst. Neural Computation, Univ. California, San Diego, Tech. Rep. 9305, 1993.
[19] M. Golubitsky and V. Guillemin, *Stable Mappings and Their Singularities*. New York: Springer-Verlag, 1973.
[20] N. E. Steenrod, *The Topology of Fiber Bundles*. Princeton, NJ: Princeton Univ. Press, 1951.
[21] C. Nash and S. Sen, *Topology and Geometry for Physicists*. New York: Academic, 1983.
[22] J. Burdick, "On the inverse kinematics of redundant manipulators: Characterization of the self-motion manifolds," in *Proc. 1989 IEEE Int. Conf. Robotics Automat.*, pp. 264–270.
[23] P. Wenger, "New results on the kinematics of robots: Generalizations of the aspects and classification of robot geometries," Laboratoire d'Automatique de Nantes, Tech. Rep. 91.21, 1991.
[24] C. W. Wampler II, "Inverse kinematic functions: A tutorial," in *Proc. IEEE Int. Conf. Robotics Automat.*, Tutorial Notes, Scottsdale, AZ, 1989.
[25] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Machine Syst.*, vol. MMS-10-2, pp. 47–53, 1969.
[26] J. Baillieul, "Avoiding obstacles and resolving kinematic redundancy," in *Proc. IEEE Int. Conf. Robotics Automat.*, San Francisco, 1986, pp. 1698–1704.
[27] C. Klein and C.-H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 3, 1983.
[28] Y. Nakamura and H. Hanasufa, "Optimal redundancy control of robot manipulators," *Robotics Res.*, vol. 6, no. 1, pp. 32–42, 1987.
[29] T. Yoshikawa, "Manipulability of robotic mechanisms," in *Proc. 2nd Int. Symp. Robotics Res.*, Kyoto, 1985, pp. 91–98.
[30] D. P. Martin, J. Baillieul, and J. Hollerbach, "Resolution of kinematic redundancy using optimization techniques," in *Proc. Amer. Contr. Conf.*, Atlanta, 1988.
[31] H. Asada and J.-J. Slotine, *Robot Analysis and Control*. New York: Wiley, 1986.
[32] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task-space augmentation and task-priority strategy," *Int. J. Robotics Res.*, vol. 10, no. 4, pp. 410–425, 1991.
[33] H. K. Khalil, *Nonlinear Systems*. New York: Macmillan, 1992.
[34] J. Baillieul, "Kinematic programming alternatives for redundant manipulators," in *Proc. IEEE Int. Conf. Robotics Automat.*, St. Louis, 1985, pp. 722–728.
[35] H. J. Ritter, T. M. Martinez, and K. J. Schulten, *Neural Computation and Self-Organizing Maps*. Reading, MA: Addison-Wesley, 1992.
[36] S. Lee and R. M. Kil, "Bidirectional continuous associator based on Gaussian potential function network," in *Proc. 1989 IJCNN*, Washington, D.C..
[37] R. L. Williams, "Inverting a connectionist network mapping by back-propagation of error," in *Proc. 8th Conf. Cognitive Sci. Soc.*, 1986.
[38] J. Kindermann and A. Linden, "Inversion of neural networks by gradient descent," *Parallel Comput.*, vol. 14, pp. 277–286, 1990.
[39] C. von der Malsburg, "Self-organized of orientation sensitive cells in the striate cortex," *Kybernetik*, vol. 14, pp. 85–100, 1973.
[40] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.*, vol. 43, pp. 59–69, 1982.
[41] H. J. Ritter, T. M. Martinetz, and K. J. Schulten, "Topology-conserving maps for learning visuo-motor-coordination," *Neural Networks*, vol. 2, pp. 159–168, 1989.

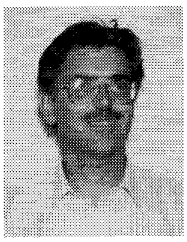
[42] J. A. Kangas, T. Kohonen, and J. T. Laakson, "Variants of self-organizing maps," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 93–99, 1990.
[43] R. Durbin and D. Willshaw, "An analogue approach to the traveling salesman problem using an elastic net method," *Nature*, vol. 326, pp. 689–691, 1987.
[44] J. Hertz, A. Krogh, and R. Palmer, *An Introduction to the Theory of Neural Computation*. Reading, MA: Addison-Wesley, 1991.
[45] D. DeMers and K. Kreutz-Delgado, "Global regularization of inverse kinematics for redundant manipulators," in *Advances in Neural Information Processing Systems*, S. J. Hanson, J. D. Cowan, and C. L. Giles, Eds. San Mateo, CA: Morgan Kaufmann, vol. 5, 1993, pp. 255–262.
[46] D. R. Baker, "Some topological problems in robotics," *Math. Intelligencer*, vol. 12, no. 1, pp. 66–76, 1990.
[47] H. Seraji, "Configuration control of redundant manipulators: Theory and implementation," *IEEE Trans. Robotics Automat.*, vol. 5, no. 4, pp. 472–490, 1989.



David DeMers received the B.S. degree in mathematical sciences with distinction from Stanford University, California, the J.D. and M.B.A. degrees from the University of California at Los Angeles, and the M.S. and Ph.D. degrees in computer science from the University of California at San Diego (UCSD).

Currenlty he is a Postdoctoral Fellow in the Department of Electrical and Computer Engineering and an Instructor with the Department of Computer Science at UCSD. He is Cofounder of OKbridge, a distributed fault-resistant real-time interactive commercial global bridge club on the Internet. His research interests include redundant manipulator kinematics, intelligent sensory-motor control, image processing and face recognition, multimedia database indexing, and retrieval and nonlinear dimensionality reduction.

Dr. DeMers is a member of ACM and the State Bar of Caliornia. He has received fellowships from the California Space Institute, the McDonnell-Pew Foundation for Cognitive Neuroscience, and NSF.



Kenneth Kreutz-Delgado (S'79–M'84–SM'93) received the B.A. and M.S. degrees in physics and the Ph.D. degree in engineering systems science, all from the University of California at San Diego (UCSD).

Currently, he is an Associate Professor of Robotics and Machine Intelligence in the UCSD Electrical and Computer Engineering Department, and a Member of both the UCSD Institute for Neural Computation and the California Space Institute. Before joining the faculty at UCSD, he was a Researcher at the NASA Jet Propulsion Laboratory, California Institute of Technology, where he worked on the development of intelligent telerobotic systems for use in space exploration and satellite servicing and repair. His interest in sensor-based intelligent systems that can function in unstructured and nonstationary environemnts is the basis for his current research activities in real-time adaptive sensory-motor control, real-time recognition/classification from multiple-time series data sets, multibody systems theory (dynamics and kinematics), and in the development of "emergent" intelligent behavior from distributed complex systems.

Dr. Kreutz-Delgado is a member of the AAAS and the IEEE Robotics, SMC, Control, and Computer Societies, and is a past Technical Editor for the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION. In 1990 he received an NSF Presidential Young Investigator Award.