IJCNN'93-NAGOYA

PROCEEDINGS OF 1993 INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS

VOLUME 3
OF 3

NAGOYA CONGRESS CENTER
OCTOBER 25-29, 1993, JAPAN

Co-sponsored by

Japanese Neural Network Society (JNNS)

International Neural Network Society (INNS)

The Society of Instrument and Control Engineers (SICE)

Chubu Bureau of International Trade and Industry

City of Nagoya

Chubu Economic Federation

The Chubu Industrial Advancement Center

IEEE Neural Networks Council (NNC)

ENNS *European Neural Network Society (ENNS)*

The Institute of Electronics, Information and Communication Engineers (IEICE)

Aichi Prefecture

Nagoya Chamber of Commerce and Industry

Nagoya Industrial Science Research Institute

Run-Time Robot Planning

V. Sanguineti†, P. Morasso†, and T. Tsuji‡

† *Department of Informatics, Systems, and Telecommunications, Genoa University, Italy*

‡ *Faculty of Engineering, Hiroshima University, Japan*

1 Introduction

Robot planning models based on real or artificial potential fields are a powerful computational metaphor [5, 11, 10, 3]. In particular, we developed a neural network architecture [7] which learns a forward model [4] of a redundant manipulator (via self-supervised training) as a map of normalized radial basis neurons and inverts the model by means of run-time gradient descent of a task-related potential field. In this paper, we propose a distributed model for the computation of the field, which is consistent with the model-inversion map, and we discuss the problem of self-synchronization between the gradient-descent process and a process for the generation of virtual trajectories of the end-effector.

2 Run-time inversion of a self-organized forward model

Let a redundant manipulator be described by a vector of joint angles $\mathbf{q} \in Q \subset R^n$, an end-effector vector $\mathbf{x} \in X \subset R^6$ (with $n > 6$), and the corresponding forward kinematic model $\mathbf{x} = \mathbf{x}(\mathbf{q})$. We approximate such a model with a a single-layer map or *neural field* F of M processing elements ($PE_i, i = 1, 2...M$) which operate in parallel receiving the common input vector $\mathbf{q}$ and reacting with a normalized Gaussian or *softmax* activation function [6, 2]:

$$U_i(\mathbf{x}) = \frac{G(\|\mathbf{x} - \tilde{\mathbf{x}}_i\|)}{\sum_j G(\|\mathbf{x} - \tilde{\mathbf{x}}_j\|)} \tag{1}$$

(The $G(\cdot)$'s are Gaussian functions of equal variance and the norm is L_2). PE_i's have limited receptive fields, centered around *preferred vector prototypes* $\tilde{\mathbf{x}}_i$'s, where the activation function peaks. The distribution of activities on the field for a given input pattern is also known as coarse or population code of that pattern. Learning is performed by means of self-supervised soft competitive learning:

$$\begin{aligned} \Delta\tilde{\mathbf{q}}_i &= \eta_1 (\mathbf{q} - \tilde{\mathbf{q}}_i) U_i(\mathbf{q}) \\ \Delta\tilde{\mathbf{x}}_i &= \eta_2 (\mathbf{x} - \tilde{\mathbf{x}}_i) U_i(\mathbf{q}) \end{aligned} \tag{2}$$

which is based on self-generated pseudo-random patterns $(\mathbf{x}, \mathbf{q})$ and carries out a smooth distribution of prototype vectors on the neural field with optimal statistical properties[1] The forward model is then approximated by the following formula:

$$\mathbf{x} = \mathbf{x}(\mathbf{q}) \approx \sum_i \tilde{\mathbf{x}}_i U_i(\mathbf{q}) \tag{3}$$

which was demonstrated [9] to be a minimum-variance estimator. This kind of estimator is also applicable to any smooth function of $\mathbf{x}$ and in particular to an artificial potential field $\varepsilon = \varepsilon(\mathbf{q})$ which represents the task constraints:

$$\varepsilon(\mathbf{q}) \approx \sum_i \tilde{\varepsilon}_i U_i(\mathbf{q}) \tag{4}$$

[1] The learning rule can be derived by minimizing the cross-entropy between the probability density function of $\mathbf{x}$ and its approximation by means of a Gaussian mixture, with the Gaussian centers in $\tilde{\mathbf{x}}_i$'s [1].

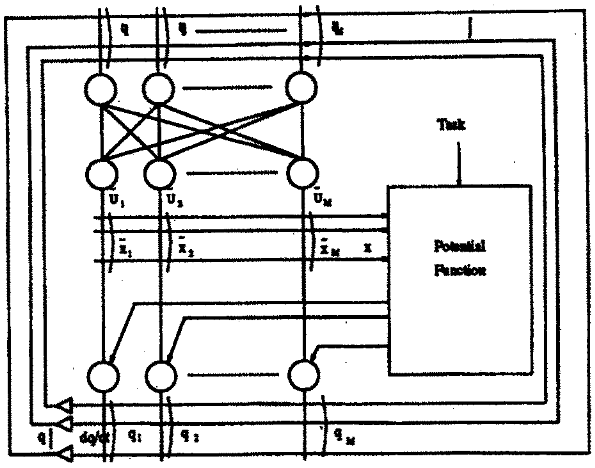

Figure 1: Block diagram of the gradient-descent network.

where the $\tilde{\varepsilon}_i$'s are samples of the potential field which are assigned to each PE in relation to its preferred sensory-motor pattern $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{q}}_i)$. Model inversion via gradient-descent exploits the following result [7]:

$$\nabla \varepsilon(\mathbf{q}) \approx -\sum_i (\mathbf{q} - \tilde{\mathbf{q}}_i)\tilde{\varepsilon}_i U_i(\mathbf{q}) \tag{5}$$

thus yielding an explicit local dynamic equation for each PE in the map:

$$\dot{\mathbf{q}} = \alpha \sum_i (\mathbf{q} - \tilde{\mathbf{q}}_i)\tilde{\varepsilon}_i U_i(\mathbf{q}) \tag{6}$$

The block diagram of figure 1 summarizes the simple feedback which allows the cortical map to carry out gradient-descent. In order to support run-time planning, the computational mechanism described above must be complemented by two additional mechanisms which are described in the two following sections: (i) a distributed mechanism for the generation of the potential field and (ii) a synchronizable mechanism for the generation of virtual targets.

3 Network model for the generation of the potential field

The use of potential fields is a powerful technique for representing task constraints of different nature and defined in different coordinate frames. In particular, we hypothesize a finite repertoire of general-purpose cost functions $\varepsilon_1 = \varepsilon_1(\mathbf{q}), \ldots, \varepsilon_N = \varepsilon_N(\mathbf{q}))$ which operate either as *attractors* (for task-components which assign a *credit* proportional to the distance from desired states) or *repulsors* (for task-components which assign a *penalty* proportional to the distance from dangerous states). For example, an attractive target-potential can be defined as follows: $\varepsilon^{tgt} = \varepsilon^{tgt}(\mathbf{q}) = \frac{1}{2} \| \mathbf{x}_T - \mathbf{x}(\mathbf{q}) \|^2$, where $\mathbf{x}_T$ is the target position. A repulsive obstacle potential can be written as: $\varepsilon^{obs} = \varepsilon^{obs}(\mathbf{q}) = f(\| \mathbf{x}_{obs} - \mathbf{x}(\mathbf{q}) \|)$, where $\mathbf{x}_{obs}$ is the obstacle point which is closest to the end-effector and $f(\cdot)$ is a monotonic decreasing function. Then, we can exploit the additivity of potential fields in order to integrate the different task-components:

$$\varepsilon = \varepsilon(\mathbf{q}) = \sum_i g_i \varepsilon_i(\mathbf{q}) \tag{7}$$

where the g_i's are relative gain coefficients which can be set according to a high-level *attentional module*.

We propose a distributed mechanism which implements the global field concept formulated above (see fig. 2 for a block diagram). It consists of a number of *potential-networks*, one for each potential function of the repertoire, which have the same size of the gradient-descent network. The generic PE (element i of network k) has two types of vector inputs: (i) the pair of prototype vectors $(\tilde{q}_i, \tilde{x}_i)$ which come from the

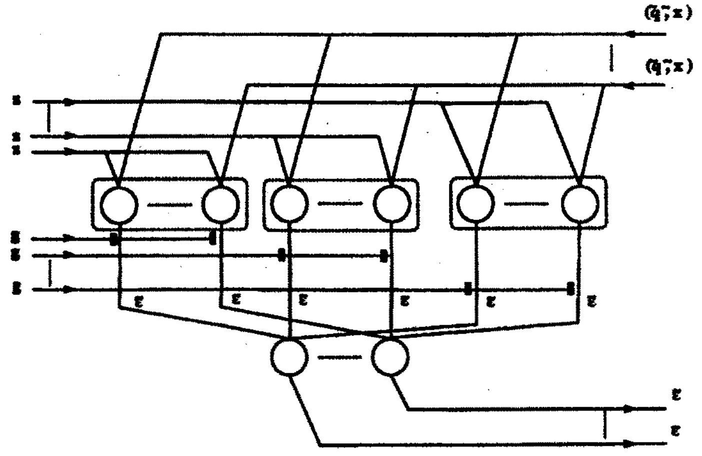

Figure 2: Block diagram of the potential-field networks.

corresponding PE of the gradient-descent network and (ii) a task-specific vector $\mathbf{z}^k$, which is common to all the PE's of the same network and is coming, as the attention coefficients, from the high-level part of the planner. The output is just a scalar ($\tilde{\varepsilon}_i^k$) which estimates the credit/penalty assigned to the input pattern according to specific task-component. For example, in the case of the target-potential, considered above, $\mathbf{z}^{tgt} = \mathbf{x}_T$ and the activation function of each PE is simply a Euclidean distance between $\mathbf{x}_T$ and $\tilde{x}_i$. The outputs of the homologous PE's of the different potential-networks are added and the global potential value is fed back to the gradient-descent network.

4 A synchronizable mechanism for the generation of virtual targets

Real-time gradient-descent requires that the potential field is incrementally updated in order to always keep the gradient-descent mechanism operating near equilibrium (local-incremental gradient-descent). This can be obtained by a target generation mechanism that smoothly shifts a virtual target $\mathbf{x}_v = \mathbf{x}_v(t)$ from the initial hand position $\mathbf{x}_0$ to the terminal target position $\mathbf{x}_T$ thus shaping, via the corresponding potential network, a target-potential field whose equilibrium state smoothly shifts the position of the end-effector along the target path. The other, overlapped potential fields introduce a sort of bias which, for the same target motion, determines task-consistent arm-motions in the null-space of the forward kinematic function.

The general requirement for a target-generation mechanism is to produce *smooth* trajectories such as trajectories with a bell-shaped velocity profile which are known to (approximately) minimize jerk. In another paper [8] we proposed a model of this kind which is based on a *time base generator* expressed as a non-linear dynamical system of the following type:

$$\dot{\xi} = \gamma[\xi(1-\xi)]^e \tag{8}$$

where ξ is a normalized scalar variable, the exponent e must be less than 1 in order to guarantee a finite duration, and the coefficient γ is proportional to the peak velocity. The trajectory of the virtual target is derived from the time base generator with a simple linear operator:

$$\mathbf{x}_v = \mathbf{x}_v(t) = \mathbf{x}_0 + (\mathbf{x}_T - \mathbf{x}_0)\xi(t) \tag{9}$$

Let us suppose that a nominal value $\gamma = \hat{\gamma}$ is chosen according to a desired duration of the movement. Then the time base generator can be started, generating a time-varying potential field $\varepsilon^{tgt} = \varepsilon^{tgt}(t)$ which excites the gradient-descent network. In general, we wish that this network tracks as precisely as possible the virtual target, i.e. we wish that at any time-instant $\varepsilon^{tgt}(t)$ is as small as possible. On the other hand, if the motion of the target potential is sufficiently slow (i.e. if γ is sufficiently small) then it is always possible to obtain any kind of positional precision. Thus, we have two contrasting requirements: timing precision

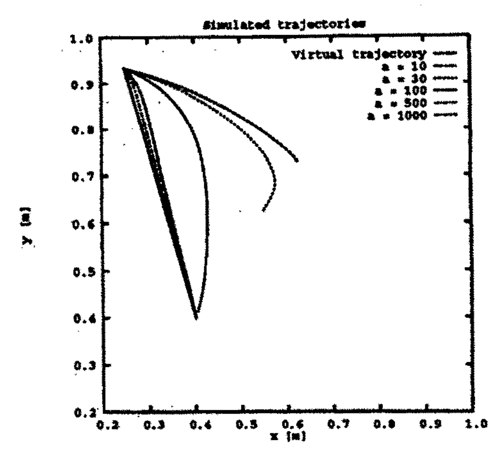

Figure 3: Simulated trajectories for different values of the gradient gain α

(measured by $\gamma - \hat{\gamma}$) vs spatial precision (measured by ε^{tgt}). The synchronization problem can then be formulated as a trade-off between the two requirements. In analytic terms, this is a very complex problem. We simply performed a preliminary simulation study, examining the effect of different values of the gradient gain α on the tracking error, without any synchronization mechanism, for a planar arm with 2 degrees of freedom. The results are reported in Fig. 3. We are currently investigating a synchronization strategy based on modulating the speed factor $\hat{\gamma} - \gamma$ as a function the tracking error ε^{tgt}.

References

[1] M. Benaim and L. Tomasini. Competitive and Self-Organizing algorithms based on the minimization of an information criterion. In T. Kohonen, K. Makisara, O. Simula, and J. Kangas, editors, *Artificial Neural Networks*, Amsterdam, 1991. North-Holland.

[2] J.S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fogelman Soulie and J. Herault, editors, *Neurocomputing*, volume NATO ASI F-68, pages 227–236. Springer Verlag, Berlin, 1989.

[3] P. Franchi, P. Morasso, G. Vercelli, and R. Zaccaria. Integrating force-field methods in a robot planning/programming language. In *Proceedings IEEE Fifth International Conference on Advanced Robotics*, Pisa, Italy, June 1991.

[4] M. I. Jordan and D. E. Rumelhart. Internal world models and supervised learning. In L. Birnbaum and G. Collins, editors, *Machine Learning: Proceedings of the Eighth International Workshop*, San Mateo, CA, 1991. Morgan Kaufmann.

[5] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5:90–99, 1986.

[6] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1:281–294, 1989.

[7] P. Morasso, V. Sanguineti, and T. Tsuji. Neural architecture for robot planning. In *International Conference on Artficial Neural Networks*, Amsterdam, 1993. in press.

[8] V. Sanguineti, T. Tsuji, and P. Morasso. A dynamical model for the generation of curved trajectories. In *International Conference on Artficial Neural Networks*, Amsterdam, 1993. in press.

[9] D. F. Specht. Probabilistic neural networks. *Neural Networks*, 3:109–118, 1990.

[10] R.B. Tilove. Local obstacle avoidance for mobile robots based on the method of artificial potentials. In *Proceedings IEEE Conf Robotics and Automation*, pages 566–571, Cincinnati, Ohio, 1990. May 13-18.

[11] C.W. Warren. Global path planning using artificial potential fields. In *Proceedings IEEE Conference Robotics and Automation*, pages 316–321, Scottsdale, 1989.