Connectionist Networks for Learning Coordinated Motion in Autonomous Systems

Jahir Pabon and David Gossard
Massachusetts Institute of Technology
77 Massachusetts Avenue Room 3-449
Cambridge, MA 02139

Abstract

A central problem inherent to autonomous systems is the absence of an external reference frame in which sensory inputs can be interpreted. It is hypothesized that, in natural systems, sensory information is transformed into a consistent internal representation that serves as an *internal invariant reference frame*. This paper presents a hierarchical connectionist network for learning coordinated motion in an autonomous robot. The robot model used in the adaptation studies consists of three subsystems: an eye-like visual receptor, a head, and an arm. The network contains a hierarchy of adaptive subnetworks for processing sensory information.

The performance of the hierarchical system was observed to improve towards an asymptotic value. The performance was found to be one order of magnitude better than that of non-hierarchical systems. This suggests that the intermediate layers may be serving as an internal invariant reference frame for the robot.

1. Introduction

Autonomous systems is a research area of large practical importance. Autonomous robots have the potential to play an important role in factory automation as well as in unmanned missions for space and undersea exploration. Autonomous systems also provide a vehicle for the study of natural, i.e. human, systems. Such research has two potential benefits: increased insight into perception and control strategies used by humans, and the identification of subsets of those strategies that can be efficiently implemented in autonomous robots.

A central problem inherent to autonomous systems is the absence of an external reference frame in which sensory inputs, such as positions, can be interpreted. In particular, an autonomous system must be able to interpret sensory information in a way which takes into account the relative positioning of sensory and motor components with respect to the system's external environment. Sensory information must be transformed into a consistent internal representation that serves as an *internal invariant reference frame*. Furthermore, it is desirable that the system *learn* this representation in order to compensate for unforeseen changes in the environment or in the system itself following growth or damage.

This paper reports on a hierarchical system of parallel distributed processing elements for producing coordinated motion in an autonomous robot. We consider it very important for an autonomous system to be able to "close the motor coordination loop" adaptively. Closing the loop involves two processes: Learning to form an internal reference frame on which sensory input can be consistently interpreted, and learning to use that internal interpretation to generate voluntary movement. Simulation studies demonstrate that the system learns an internal invariant reference frame and uses that internal reference frame to produce coordinated motion.

2. Related Work

Most of the research work in adaptive control techniques for robotic applications (Koivo and Guo, 1983; Leininger, 1984; Dubowsky and Kornbluh, 1985; Atkeson and McIntyre, 1986; Slotine and Li, 1987) is based on the use of some external reference frame to measure performance errors of the system and therefore not directly applicable to the study here.

Networks of parallel distributed processing elements, i.e. "neural networks", possess a number of useful computational properties (Grossberg, 1988; Hopfield, 1982; Kohonen, 1984; Rumelhart, 1986). Several preliminary studies have partially shown that these networks have the potential to accomplish adaptation in autonomous systems (Barto, 1984; Bullock, 1988; Grossberg and Kuperstein, 1986; Kawato, 1987; Kuperstein, 1987; Pabon and Gossard, 1987b; Psaltis, 1987).

Kuperstein (1987) presented a connectionist model that adaptively controls a visually guided robot arm to reach target spots in three dimensions. The visual input in his model, however, was produced by two cameras preprogrammed to point at the target. This, in essence, provided an external reference frame for the system. The camera orientations were subsequently encoded into the activation of two two-dimensional arrays of units. These activation maps were then used as the input to the adaptive network that sent signals to the arm actuators to move the arm endpoint to the desired position.

Kuperstein makes reference to his previous work (Grossberg and Kuperstein, 1986) on adaptive control of saccadic eye movements as a possible mechanism to close the loop, but no model of the complete process was presented. In addition, no mathematical support for the convergence of the adaptation method was presented.

3. Neural Networks: A Short Review

Connectionist networks are arrays of simple, neuron-like, highly interconnected computing elements. One of the basic network architectures is the two-layer feed-forward network. Figure 1 shows the topology of this network. The network consists of a set of input units, **x** , connected to a set of output units, **y** , through a set of weights, **w**. The activation of the output units in the network is given by

$$y_i = f(\sum_{j=1}^{n} w_{ij} x_j)$$

where x_j is the activation of unit j connected to unit y_i, w_{ij} is the strength (weight) of the connection from unit x_j to y_i, and f is an output activation function.

Adaptation in these networks is achieved by regulating the strength of the connections (weights) among the network units. One well-known adaptation method is that of backpropagation (Rumelhart, 1986). During the learning process, the weights w_{ij}, are modified so as to minimize the difference between the output activation y_i, and a reference output r_i. This is achieved by using the expression:

$$\delta w_{ij} = \alpha x_j (r_i - y_i) (f')$$

where δw_{ij} is the change in the weight w_{ij}, α is the learning rate parameter, and (f') is the derivative of the output activation function. This adaptation law is frequently referred to as the *delta rule*. The learning parameter, α, determines the rate and performance of the adaptation process. It should be emphasized that this network is just one building block in the control structure, and that the reference signal, r_i, although external to the network, is produced by components internal to the autonomous system that form part of the same structure.

Two-layer feed-forward networks are useful as building blocks in applications involving adaptive mappings. The application in question question here is that of *Learning Motion Control*. In this application, the input units encode commands and sensory signals; the output units produce motor signals that are fed to plant actuators to generate motion; and the reference signals represent desired plant responses.

4. Model and Control

Figure 2 presents the robot model used for our adaptation studies. It consists of three hierarchical subsystems: an eye-like visual receptor, a head, and an arm. Each subsystem has two degrees of freedom: the eye subsystem can rotate in two directions, the head subsystem can translate in two directions, and the (planar) arm has two links with rotational joints.

In each degree of freedom, the position is controlled by an antagonist pair of muscle-like actuators, i.e. opposing springs whose stiffness is regulated by control (activation) signals. In the eye, for example, changing the activation signals to a pair of actuators causes a rotation of the eye to a new position where the spring forces are in equilibrium. See Figure 3.

The eye contains a population of light receptor units arranged in a two-dimensional array, called here the *retina*. The level of activation of each receptor is determined by the amount of light incident upon it. Thus a target light spot impinging upon the retina generates a distribution of activations across the units in which the most active units will be those closest to the point where the light strikes the receptor array. This distribution of activation is called the *retinal map*. In this study a decaying exponential (gaussian) distribution was assumed and is described in the Appendix. An on-center off-surround receptive field similar to those in human retinal receptors could also be used and would produce similar results.

The set of activation signals sent to the eye and head actuators were similarly encoded into 2D arrays of units. These arrays are called the *eye position map* and the *head position map* respectively.

The protocol for the learning experiments was the following. During the learning phase, the current endpoint of the arm is used as the target. A random signal generator is used to supply activation signals to the arm actuators so as to span the complete arm workspace. The system's goal is to use the sensory information (retinal map, eye position map, and head position map) to generate command signals to the arm actuators which match those produced by the random generator, so as to keep its endpoint in the original position. The distance between desired and actual arm endpoint positions is taken as the error, a measure of the system performance. Learning is assumed to be complete when the average error over the arm's workspace is sufficiently small (e.g., less than 5% of the characteristic length of the workspace). After this, a testing phase can be carried out with visual targets presented to the system in the form of light spots on the viewing plane.

The adaptive control scheme used is presented in Figure 4. The thick lines denote that the given signal is encoded onto a population of units (a 2D array).

Given current eye and head positions (centered at the beginning of the learning process), the target generates a retinal map on the visual receptor array. The retinal map and the current eye position map are input to a first adaptive network (number 1 in Figure 4). The output of this first network evolves, during learning, into a representation of the target that is invariant with respect to eye orientation, i.e. given a fixed target and head position, this signal remains constant independent of changes in orientation of the eye. This signal is called TPME (for Target Position Map invariant with respect to Eye orientation). The TPME is then input to a second adaptive network (number 2 in Figure 4) which generates the motor commands to the eye actuators.

After the eye reaches its new orientation, an error signal is generated by the retinal unbalance decoder, which weighs the eccentricity of the retinal map. The error signal is used to modify the connectivity matrix of network 2 using the delta rule. Backpropagation is then used to modify the connections in network 1. A simple network architecture to measure the eccentricity of an activation map is described in (Pabon, 1987a).

The TPME and the head position map, are used as input to a third network (number 3 in Figure 4). The output of this network evolves, during learning, into a representation of the target that is invariant with respect to head position, i.e. given a constant target, this signal remains constant independent of changes in orientation of the eye or position of the head). This signal is called TPMH (Target Position Map invariant with respect to Head position). The TPMH is then input to a fourth adaptive network (number 4 in Figure 4) which generates the motor commands to the head actuators. After the head reaches its new position, an error signal, obtained by weighing the eccentricity of the eye position, is used to modify the connectivity matrix of network 4 according to the delta rule. Backpropagation is again used to modify the connections in network 3.

The TPMH is also used as input to a fifth network (number 5 in Figure 4). The outputs from network 5 are the command signals to the arm actuators. The arm will then move, attempting to reach the target. During learning, the output from network 5 is compared to the random signal that originated the arm movement. The difference is then used to modify the connectivity matrix of network 5.

5. Results

A number of simulation studies was conducted to examine the qualitative and quantitative behavior of the model and its control. In the first study the eye subsystem alone was examined. The objective was to select appropriate values for the model parameters. The results from this study were presented in (Pabon and Gossard, 1987b), where it was found that the values of the learning rate parameter, α, proposed by the authors were always in reasonable agreement with the best values derived from the simulations.

5.1 Entire System with Internal Layers

The results from the eye simulations were used in a second study, where the eye, head and arm subsystems were examined working together as proposed in Figure 4. The parameter values used in the simulations of the entire system were the following (length values are normalized and therefore nondimensional: eye radius, 0.25; distance from eye center of rotation to the viewing plane, 3.0; head workspace, square of dimension 4*4 centered about the base joint of the arm; arm links length, 1.0; arm workspace, defined by the joint limiting angles (0°, 135°); retina composed of a square array of 5*5 receptor units; eye position encoded into a square array of 5*5 units; head position encoded into a square array of 5*5 units; encoding parameter, 30 for all maps; output activation function, $(1-e^{-x})/(1+e^{-x})$;

The error of the arm (i.e. the global error) is defined as the distance between the target point and the actual arm endpoint position, expressed as a percentage of the arm link length. The time history of the error of the arm position is shown in Figure 6(a)-(b). It can be seen that the performance of the system approaches an asymptotic value. The steady state error, defined as the average error over the last 10% of a run of 10^4 iterations, was 8.4% of the arm link length

5.2 Process Without Internal Layers

Simulations were also carried out of an alternative adaptive controller with no internal layers, i.e. all the sensory information was fed directly into a two-layer network which generated the motor commands to the arm. This controller is shown in Figure 5. Using the same number of units per map and a similar number of iterations, the steady state error of the arm (global error) in the model without internal layers was on the order of 50%. This is about one order of magnitude larger than the error observed in the model with internal layers.

6. Conclusions

The steady state error of the arm decreases asymptotically towards a small value (~8% of the arm's length). The asymptotic value to which the system's error tends is sensitive to several parameters: the number of units used in the encoding maps, n , the encoding parameter, s, and the learning rate parameter, α.

The number and structure of internal layers play an important role in the efficiency of the adaptation process. Specifically, the fact that the performance of the system with internal layers is so much superior to the performance of the system without internal layers suggests that the system is using networks 1 and 3 as what amount to an internal invariant representation of its environment.

The TPM-E described here (i.e. the output of adaptive network 1 in Figure 4) was inspired by experimental evidence that activation levels of certain cells in the posterior parietal cortex (of the monkey brain) are a function of both retinal maps and current eye position. The control system presented here is an extrapolation of this basic idea to handle additional degrees of freedom. The successful performance of the system studied here suggests that it is possible that a set of cells exists whose activity further correlates the activity of those cells in the posterior parietal cortex with current head position, corresponding to the TPM-H of the system here.

Parallel distributed controls for artificial systems would be very robust. The weights in the control system studied here were initialized to random values. Through learning, they eventually "encoded" the kinematics of the particular robot geometry. The same control system could, without modification, learn other robot geometries. Such systems could thus compensate for unforeseen changes in the environment or in the robot itself following growth or damage. The performance of these systems would also degrade gracefully with the loss of individual units.

Appendix A. Encoding of Continuous Variables on a Population of Units

A pair of continuous variables (σ,γ) in the ranges $(\sigma_{max},\sigma_{min})$ and $(\gamma_{max},\gamma_{min})$ respectively can be encoded as the activation of a two-dimensional set of units $\{x_{ij}\}$, i,j=1,n, using the encoding function:

$$x_{ij} = \exp\left(-s\left[\left(\frac{\sigma-\sigma_i}{\delta\sigma}\right)^2+\left(\frac{\gamma-\gamma_j}{\delta\gamma}\right)^2\right]\right)$$

where s is the encoding parameter; $\delta\sigma = \sigma_{max} - \sigma_{min}$; $\delta\gamma = \gamma_{max} - \gamma_{min}$; (σ_i, γ_j) are the characteristic values of unit ij (value of the pair (σ,γ) that produces a maximum activation of the unit).

The encoding parameter, s, determines the degree to which the activation is distributed across the units.

References

[1] Atkeson, C. G., and McIntyre, J., "Robot trajectory learning through practice", Proc. of IEEE Conf. on Rob. and Autom., 1986.

[2] Barto, A. G., Sutton, R. S., and Anderson, C. W., "Neuronlike elements that can solve difficult learning control problems". IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-14, 1984, 834-846.

[3] Bullock, D., and Grossberg, S., "Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation", submitted to Psychological Review, 95(1), 1988.

[4] Dubowsky, S., and Kornbluh, R., "On the development of high performance adaptive control algorithms for robotic manipulators", in Robotics Research: The Second International Symposium, Hanafusa, H., and Inoue, H. eds., MIT Press, Cambridge, MA, 1985.

[5] Grossberg, Stephen, "Nonlinear Neural Networks: Principles, Mechanisms, and Architectures", Neural Networks, Vol. 1, No. 1, pp. 17-61, 1988.

[6] Grossberg, S., and Kuperstein, M., Neural Dynamics of Adaptive Sensory-Motor Control, North-Holland, New York, 1986.

[7] Hopfield, J. "Neural networks and physical systems with emergent collective computational abilities". Proc. Natl. Acad. Sci. vol. 79, pp. 2554-2558, (1982).

[8] Kawato, M. , K. Furukawa and R. Suzuki, "A Hierarchical Neural-Network Model for Control and Learning of Voluntary Movement", Subm. to Biol. Cybern. 1987.

[9] Kohonen, T. Self-organization and Associative Memory. New York, Springer-Verlag, 1984.

[10] Koivo, A., and Guo, T., "Adaptive linear controller for robotic manipulators", IEEE Trans. Auto. Control, AC-28(2), pp. 162-170, 1983.

[11] Kuperstein, M. "Adaptive Visual_motor Coordination in Multijoint Robots using Parallel Architecture", Proc. of IEEE Rob. and Autom., pp. 1595-1602, 1987.

[12] Leininger, G. G., "Adaptive control of arms using self-tuning methods", in Robotics Research: The First International Symposium, Brady, M., and Paul, R. eds., MIT Press, Cambridge, MA, 1984.

[13] Pabon, J., and Gossard, D., "A two-dimensional neural network to stabilize position of input patterns". MIT CADLAB technical report. July 1987a.

[14] Pabon, J., and Gossard, D., "A methodology to select appropriate learning rate parameters in feed-forward networks", presented at the Third Annual Aerospace Applications of Artificial Intelligence Conference (AAAIC), Dayton, Ohio, October, 1987b.

[15] Psaltis, D., Sideris, A., and Yamamura, A., "Neural Controllers", Proc. of IEEE First International Conference on Neural Networks, Vol. IV, pp.551-558, San Diego, CA, 1987.

[16] Rumelhart, D., Hinton, G. and Williams, R. "Learning internal representation by error propagation". In Distributed Parallel Processing. MIT Press, Cambridge, 1986.

[17] Slotine, J.-J., and Li, W., "Adaptive robot control: A new perspective", sub. to IEEE Conf. on Decision and Control, 1987.

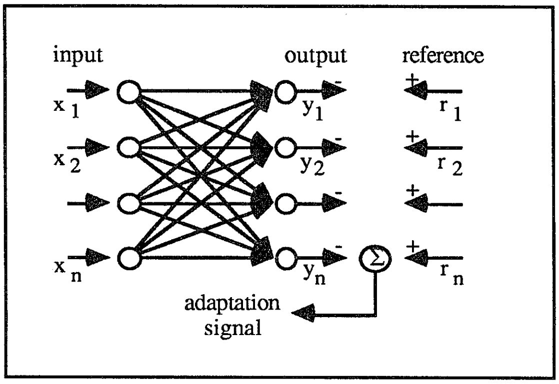


Figure 1. A two-layer feed-forward network.

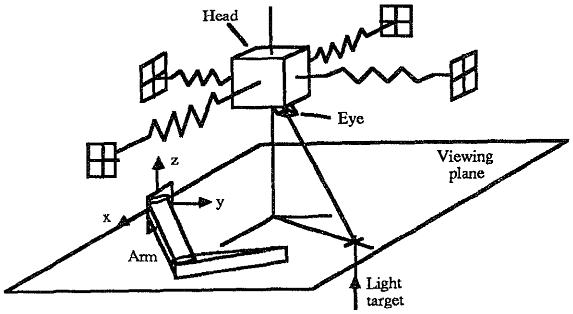


Figure 2. The Autonomous Robot.

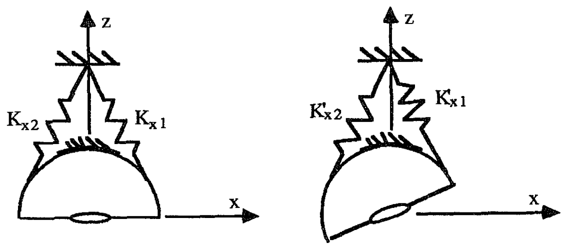


Figure 3. The eye subsystem

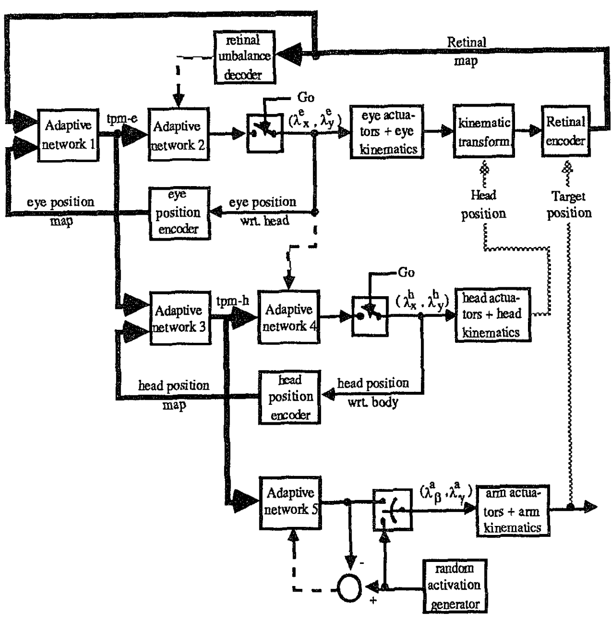


Figure 4. The adaptive control loop.

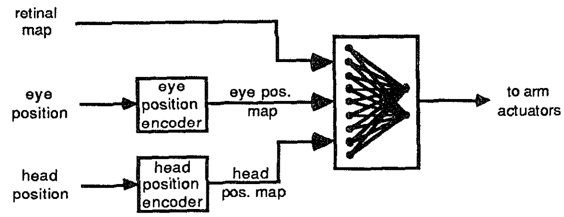
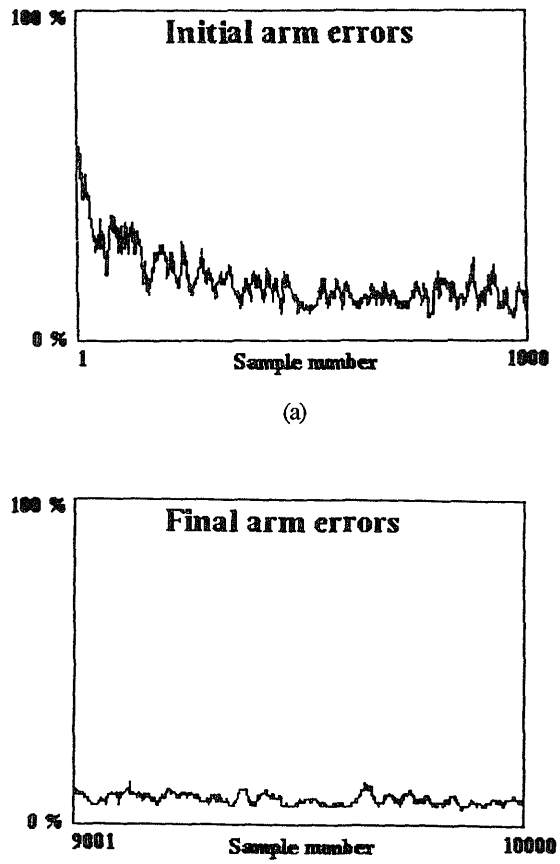


Figure 5. Adaptive Process without Internal Layers



(a)

(b)

Figure 6. Time evolution of the arm position error