

Hardware Implementation of a Semi-Autonomous Robotic Neurocontroller

Chad Cox, Kevin Priddy, Karl Mathia, Richard Saeks, Robert M. Pap

Accurate Automation Corporation
7001 Shallowford Road, Chattanooga, TN 37421

Abstract

A neural network semiautonomous robotic arm controller has been developed and implemented in a neural network PC board. The robotic controller performs end-effector path planning, inverse kinematics, and joint control to move the end-effector to a commanded position. This paper discusses the inverse kinematics and neural network hardware used to implement the controller. The inverse kinematics algorithm is based upon a Linear Hopfield Network (LHN). The neural network hardware implements this algorithm, together with the entire neural robotic controller, in real time.

1.0 Controller Design

Accurate Automation Corporation implemented neural networks for robotic control applications^{1,2,3,4,5}. This project addresses the issues in path planning, inverse kinematics, and joint (motor) controls together with the development of the neurocontrol hardware required to implement the resultant control system and interface it to real robotic arms.

The joint controller has been tested on two robot arms. These robot arms are the Extendable Stiff Arm Manipulator (ESAM) and the Proto-Flight Manipulator Arm (PFMA). Both of these arms are very different in physical attributes and motor configuration, yet the same unmodified joint controller software can control both of them. The neuro-controller is shown in block diagram form in Figure 1. The controller is based in part upon the work of Seraji⁶ and has tremendous adaptability to large payload variations^{1,2}. The neurocontroller operates at three levels. At the highest level is path planning. At the middle level is inverse kinematics. At the lowest functional level is decentralized adaptive joint control. We have developed neural networks for multiple joint controllers and inverse kinematics, and are developing a path planning network. Our design can be fed goals, either from a human, computer, or from other neural networks which may be developed in the future.

2.0 Inverse Kinematics

We have developed an solution to the inverse kinematic problem using a Linear Hopfield Network (LHN)^{7,9}. The difficulty of complex inversion or no-invertability of robot inverse kinematics is circumvented by linearization of the forward kinematics of an appropriate energy function, which is then minimized by linear dynamic networks. The results of the dynamic optimization process are the 'best' joint angle rates which minimize the manipulator's position error. We call the network the Linear Hopfield Network because of its similarities with the

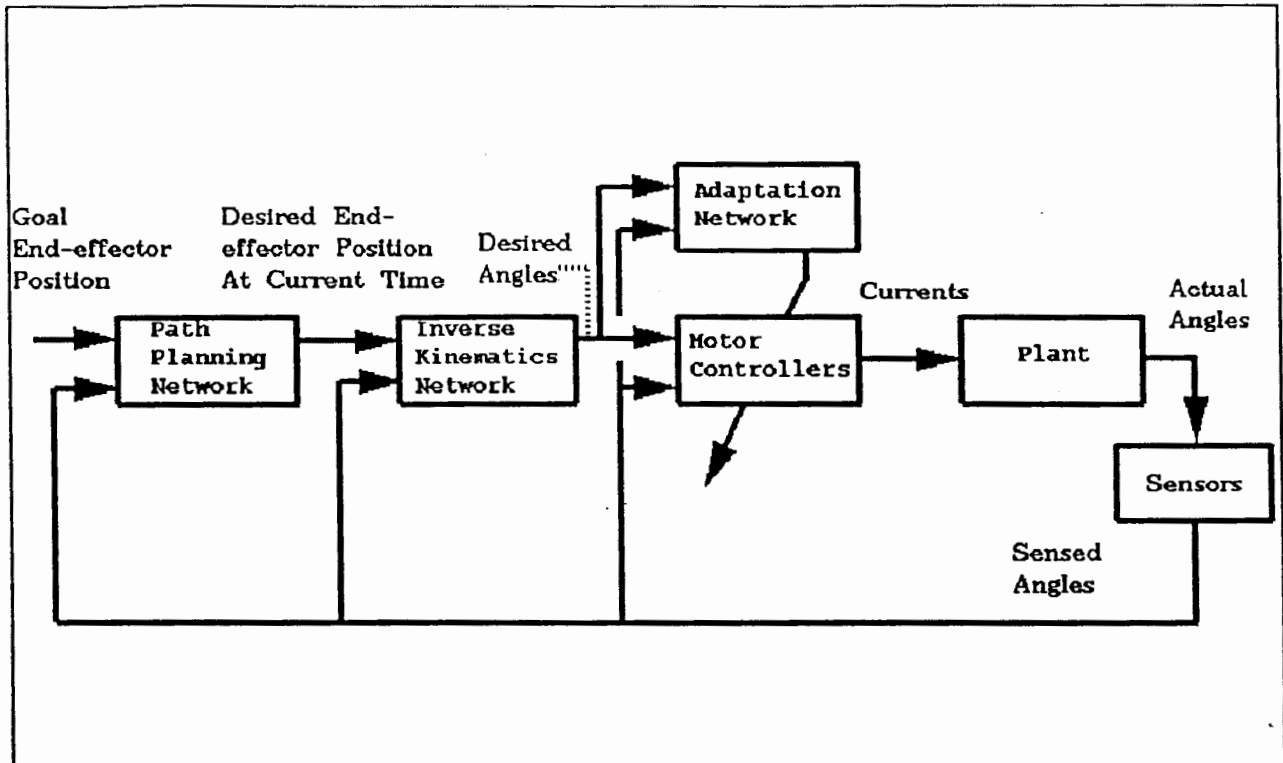


Figure 1. Block Diagram of Neural Network Robotic Controller.

original Continuous-Hopfield Network. A derivation as well as a further discussion and a convergence proof for the synchronous dynamics of the LHN were reported previously⁵. The simulated tracking control of a trajectory in 3-D Cartesian space with a three-joint robot manipulator demonstrates the performance of this approach.

The initial concept of using a neural network to circumvent a closed form solution to the inverse kinematics problem by linearizing the direct kinematics and then defining and minimizing an energy function were presented earlier⁸. To find the required joint angles for a desired manipulator position, the inverse kinematics problem is reduced to the numerical solution of a linear system. The linear system is specified and solved for each manipulator position, i.e. for each point on the manipulator's trajectory in Cartesian space. The required associated point in joint angle space is determined via an optimization process.

An appropriate energy function is represented in a form which can be minimized by a linear, fully connected, recurrent network. As opposed to the Continuous Hopfield Network, which has nonlinear transfer functions and is used as an autoassociator, the LHN has linear transfer functions and solves systems of linear equations. The LHN can be constructed to guarantee convergence to an optimal solution, here the 'best' joint angle velocity to minimize the manipulator's velocity error. The resulting discrete velocities are integrated to track the preplanned end-effector path.

3.0 AAC Neural Network Processor

We are implementing the entire robotic controller, including the inverse kinematics network, on the Accurate Automation Corp. MIMD Neural Network Processor module (AAC NNP). A block diagram of the NNP architecture is given in Figure 2. The underlying philosophy in the design of the AAC NNP has been to achieve maximum computational efficiency in both a single processor and multiprocessor environment by optimizing the design to compute neuron values - and nothing but neuron values¹⁰. Indeed, this is ideally suited to a neural network robotic control application and stands in stark contrast to previously proposed processors which are typically based on classical SIMD (single instruction multiple data) matrix/vector multiplication architectures. Our design fully exploits the intrinsic characteristics (sparse, local, random) of the neural network topology. By using an MIMD parallel processing architecture one can update multiple neurons in parallel with efficiency approaching 100% as the size of the neural network increases. To achieve the desired efficiency, we have adopted a design which:

1. uses an instruction set which is optimized for neural network processing allowing one to compute a neuron activation without arranging the weight matrix into linear arrays and/or inserting "artificial zero weighted connections",
2. uses an MIMD (multiple instruction multiple data) parallel processing architecture to permit neurons with totally different input topologies to be updated simultaneously without loss of efficiency, and
3. uses dual neuron memories to virtually eliminate memory contention and maintain absolute memory coherence.

This architecture allows us to implement a relatively simple single processor NNP module and then string together multiple NNP modules along a dedicated Interprocessor Bus with computational power (and cost) increasing "almost" linearly with the number of modules¹⁰.

The AAC MIMD Neural Network Processor:

1. is designed to implement multiple interconnected neural networks of differing architecture simultaneously using 16 bit twos complement binary fixed point arithmetic with up to 8K total neurons and 32K connection weights per module,
2. is capable of running at 134,000,000 connections (byte wide multiply / additions) per second per module for a total of a billion plus connections per second in an 8 processor array,

The neural network processor, while developed for this robotics application is capable of performing a variety of tasks. The algorithms currently implemented on the neural network board are all feedforward neural network paradigms and the robotic control neural network structure. In addition to having the ability to run trained networks, the neural network board can be "trained" with some degradation in performance. The training performance is much better than that obtained in software.

4.0 Conclusion

Accurate Automation Corporation is designing a generic robot arm neurocontroller for use in underwater and space applications. Results have been obtained for the inverse kinematics and joint control portions. The inverse kinematic portion utilizes the speed of neural networks without compromising accuracy. Simulation results show this to be a viable approach. A neural network processor has been built to implement these algorithms together in real time on an actual robotic arm. The PC version of the neural network processor has been completed and a VME version is being developed.

5.0 Acknowledgements

This work is sponsored by the Department of the Navy, Office of Naval Research, under Phase II SBIR contract N00014-92-C-0268 and OASN(RD&E), Product Assessment Division. The COTR is Dr. Joel Davis, ONR Code 342CN. The VME board is being built under NCCOSC NRaD Phase II SBIR contract N66001-90-C-7021. The COTR is Mr. Richard Akita, NRaD Code 454.

6.0 References

1. Cox, C., J. Edwards, R. Saeks, M. Lothers, R. Pap, and C. Thomas. "A Neural Network for Joint and Motor Control", World Conference on Neural Networks, Portland. Vol. 3. pp. 350-353, Lawrence Erlbaum Associates: New Jersey, 1993.
2. Cox, C., R. Saeks, M. Lothers, R. Pap, and C. Thomas. "A Neurocontroller for Robotic Applications", Proc. of the International Conference on Systems, Man, and Cybernetics, Chicago. pp. 712-716. IEEE: Salem, 1992.
3. Parten, C.R., R.M. Pap, and M.D. Lothers. "Neural Network Robotics Control", Proc. of WESCON/90. Western Periodicals: North Hollywood, 1990.
4. Parten, C.R., R.M. Pap, and C.R. Thomas. "Neurocontrol Applied to Telerobotics for the Space Shuttle", Proc. of the Inter. Neural Network Conf., Paris. pp. 229-236. Kluwer Academic Publishers: Boston, 1990.

5. Mathia, K. and R. Saeks, (1994). "Inverse Kinematics Via Linear Dynamic Networks," To be presented at *World Conference on Neural Networks*, San Diego, 1994.
6. Seraji, H. "Decentralized Adaptive Control of Manipulators: Theory, Simulation, and Experimentation." *IEEE Transactions on Robotics and Automation*. Vol 5. No. 2., pp. 183-201., 1989.
7. Hopfield, J. and Tank D. "Computing with Neural Circuits: A Model," *Science*, Vol. 233, August. pp. 625-633, 1986.
8. Guo, J. and Cherkassky, V., 1989. "A Solution to the Inverse Kinematics Problem in Robotics Using Neural Network Processing," *International Joint Conference on Neural Networks (IJCNN)*, Washington, D.C., Vol. II, pp. 299-304, IEEE: New York, 1989.
9. Maren, A., Harston, C. and Pap,R.,(1991). *Handbook of Neural Computing Applications*, Academic Press.
10. Saeks, R., Priddy, K., Pap, R., and Stowell, S., (1994). "Design of a MIMD Neural Network Processor", *Proceedings of SPIE Applications of Neural Computing V*, Orlando.