Distributed Robotics: An Intelligent System

Research Plan

Malcolm Stagg

10 December 2006

1 Purpose

In a colony of insects, each individual possesses very little of its own intelligence, but collectively the system is capable of many advanced tasks. As the colony develops over time, a form of communication and language develops, which genetically and neurologically becomes known throughout the entire group. This form of neural-optimized communication allows various individuals in the system to provide feedback and transport information to each other, while adapting to the current needs of the cluster.

In this project, a customizable design is proposed for such a system, which will be capable of performing intelligent tasks. Intelligence and abilities are distributed among various robotic units. A feedback-based multi-module neural network is designed to allow serialized learning over a common communication link. Input sensors such as miniature cell phone cameras, microphones, and accelometers will be distributed throughout the system. Information from these sensors will be processed by the neural network and communicated to the other units. Self-feedback will allow the individual modules to be aware of their own undesirable behavior, whereas group-feedback will allow a form of societal development to take place and optimized communication to develop.

An intelligent system of distributed robots has many potential uses in applications where low-cost robots are required to complete a massive task, such as military surveillance [13], mining, agriculture, and inspection of precise equipment such as aircraft. Current systems for these applications are rather limited in terms of intelligence, due to the cost of effective hardware-based neural systems. The current use of non-neural-based programs to mimic intelligence [5, 13] is inadequate in terms of being able to learn new tasks and adapt to the environment over time.

In this proposed system, cost-limitations of the distributed system are overcome by the use of an adaptive hardware-based system which may be implemented in VLSI (Very Large Scale Integration) CMOS (Complementary Metal Oxide Semiconductor) technology [1] using mixed-signal high-frequency PWM (Pulse Width Modulation) [6]. A low-cost prototype may be developed using a fully-digital stochastic FPGA (Field

Programmable Gate Array) -based system. The prototype design allows the design to be fully reconfigurable while providing a reasonable neuron density and cost per neuron. A general-purpose PWM-based implementation allows a scalable system size with excellent routability for large designs, small neurons and synapses for high density, and a low cost overhead due to the possibility of implementing the same system in many applications.

This proposed distributed system will allow complex processing tasks to take place in a mobile cluster of robots capable of performing large-scale tasks which would be difficult or impossible for a single robot [13]. Embedded hardware systems such as camera processing, object recognition, auditory processing, and position mapping may take place with the combined use of the adaptive neural network in conjunction with predetermined and dedicated processing on the FPGA. The use of such processing will improve data representation as well as the quality of object recognition and other tasks performed on the neural network.

2 Engineering Goals

2.1 To design a software simulation for a system of interacting nodes with small simulated neural networks to provide feedback-based intelligence where stimuli are presented in a simulated environment.

2.2 To construct small prototype robotic nodes using inexpensive neural networks constructed on FPGAs. Real-time hardware processing of camera inputs will also be implemented in the same design.

2.3 To analyze the development of the large-scale software- and small-scale hardware- based systems in terms of language development and ability to learn and to perform simple tasks collectively.

3 Background Information

3.1 Distributed Robotics

Distributed and swarm robotics are rather new fields of artificial intelligence where the tasks normally given to a single robotic device are distributed among a cluster of robots. This cluster is often able to better perform a large scale tedious task than a single robot. Typically these systems provide very limited “intelligence” in each device as no more than a set of instructions [5, 13].

This form of distributed intelligence works reasonably well for simple applications, but is limited in that it is not very well adaptable to changes in the environment and will not perform tasks which require higher-level processing. By combining intelligent hardware with these simple robotic devices, intelligence for more complex tasks, than can be handled with simple algorithms, is provided.

Figure 1 – An existing system of distributed robotics

3.2 Neural Networks

MLPs (Multi-Layer Perceptrons) are the most common form of neural networks. These devices consist of basic processing elements (neurons) and connections between these elements (synapses). Each synapse stores a weight value which is multiplied by the neuron output to influence the structure of the network. [4, 15]

The popular Backpropagation learning algorithm works when a source of feedback is available at the output of the neural network. This feedback will propagate backwards throughout the neural network and, over time, correct errors within the synapses. Synapse changes are made based on the calculated responsibility of error for each of the weights.

Hebbian learning, another popular method, uses a correlation between connected neurons to make changes to the synapse weight. If the values of the synapses seem to be related, the weight will increase. If they are unrelated, the weight will decrease. This allows the neural network to recognize patterns based on the input data alone, with no other source of feedback.

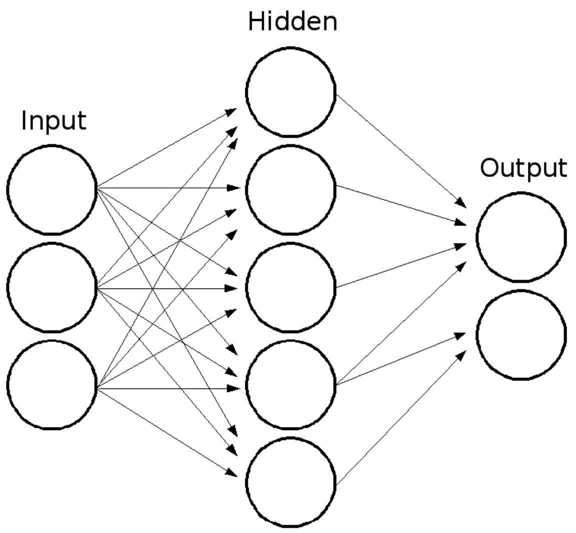

Figure 2 – Multi-Layer Perceptron Neural Network. Arrows represent synapses and circles represent neurons.

3.3 Image Processing

The SIFT (Scale-Invariant Feature Transformation) algorithm is used to identify feature-points in the video data. This algorithm works by Gaussian-filtering the image at a number of different scales and levels. Differences between neighboring scales and levels are used to determine whether each point has the potential to be a significant feature in the image. [9]

The potential feature points are further analyzed to determine the magnitude of the feature point, and the direction of the intensity gradient with respect to neighboring gradients. SIFT has an advantage over other methods in that it is largely invariant to scale and rotation changes in two-dimensions.

Implementation of this algorithm on an FPGA has some significant advantages over computer-implementation in terms of parallel processing. Many of the Gaussian kernels can run at the same time to provide an output faster than a computer would be capable. Also, an FPGA provides a direct link between the camera interface and image processing hardware.

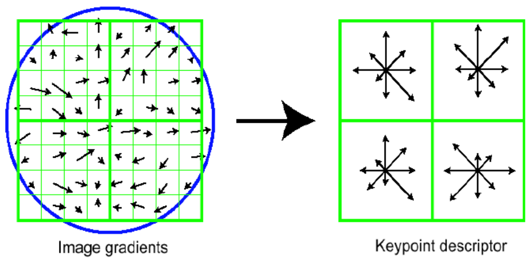

Figure 3 – Calculation of a SIFT descriptor based on gradient data [9]

The high-dimensional output of the SIFT algorithm may be used to match with feature points of known images, or further analyzed (e.g. with a neural network) for non-database object recognition.

4 Description of Methods

4.1 Neural Layout

A neural network layout has been designed to allow self- and group- feedback to take place, as well as providing a method of serialized communication (see 4.4). The preliminary design incorporates a combination of Hebbian (pattern-directed) and Backpropagation (teacher-directed) learning, to allow the device to learn data and societal patterns as well as correct errors with teacher-directed learning. In this case the teacher may be a human overseer, feedback from the rest of the colony, or self-feedback when an error is detected. [4]

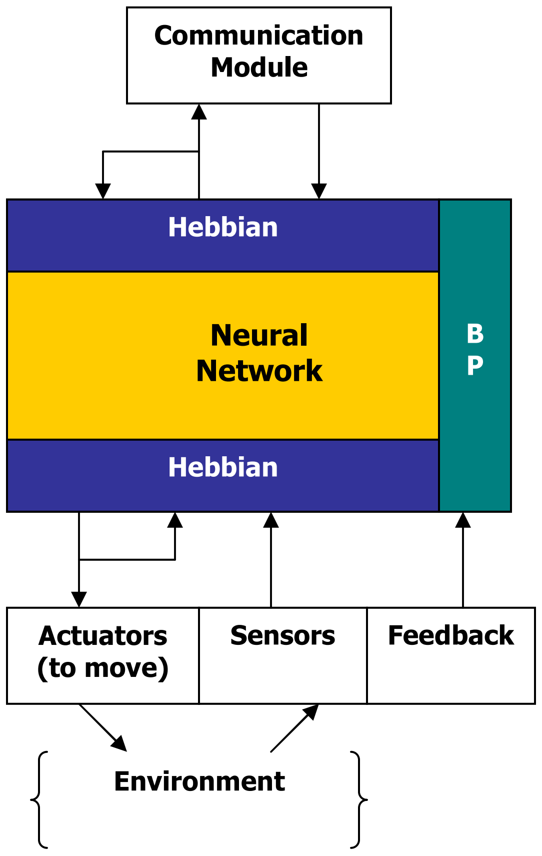

Figure 4 – Proposed neural network will combine Hebbian and Backpropagation algorithms in a closed-loop control system

Many feedback paths allow the neural system to be placed in a closed-loop where it is fully aware of, and can learn from its environment. An actual implementation of the neural network will require subdividing the network into multiple modules within each device. These modules will include:

- Communication (see **4.4**)
- Actuator control
- Feedback generation
- Object recognition (see **4.3**)
- Auditory processing
    - *(for future implementation)*

Inputs and outputs for each of these modules will only be connected to the appropriate module, but modules will share synapse connections to allow processed data to be shared throughout the device. A modular approach to the neural network allows for greater specialization for individual tasks and mimics the similar arrangement of the biological brain.

4.2 Camera Interface

One-megapixel cell phone cameras are used to provide a video input to various nodes in the system. These cameras send the image data serially at a rate of approximately 15 frames per second using the SMIA format. While the expected image quality from these cameras is rather low, it will provide an inexpensive high-resolution image of the surrounding environment which each camera-equipped node can use for advanced tasks such as feature identification, object recognition, and mapping.

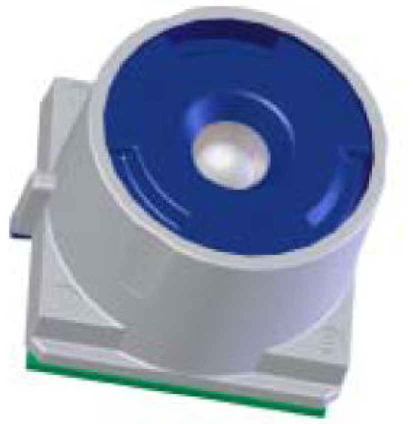

Figure 5 – Small 1 MP cell phone camera module [17]

The interface overhead for these devices is rather low, requiring only a UART (Universal Asynchronous Receiver Transmitter) which may be implemented in the FPGA, and a RAM buffer of at least 1MB to store the image for processing. In this system, a larger RAM buffer will be used to provide faster image processing and filtering.

4.3 Image Processing

FPGA-based real-time image processing will implement a method based on the SIFT (Scale Invariant Feature Transformation) algorithm to identify feature-points in the video data. These invariant features will be able to be presented to the neural network for some form of object recognition to take place, and/or transmitted to other devices for group navigation or mapping tasks.

The keypoints generated by SIFT are reduced to low dimensionality and then presented invariantly to the object recognition module of the neural network. At this point, recognition will be attempted if the object is known, and object data may be shared with other devices. If the object is recognized remotely but not locally, feedback may be given over the communication channel from another device.

SIFT is implemented in FPGA hardware by creating groups of Gaussian kernels which are multiplied by the incoming image data. Features are located based of the DoG (Difference of Gaussian) values at each position of the image.

Figure 6 – The SIFT algorithm used for object recognition in an existing database system [3]

4.4 Inter-Device Communication

While there are several possible methods of inter-device communication, radio communication seems to be the most advantageous [13, 18, 19] possibility. This will allow any units within a certain radius to communicate with each other very easily, and the possibility for other nodes to act as repeaters when the recipient is out-of-range.

One potential problem with most forms of communication is addressing: the ability for the message to get from the sender to the intended recipient. Experimentation will take place to find out if the colony is capable of creating its own method of communication; however several structures, including a basic method of addressing, will be fixed in hardware. It is still possible, though, for the wrong recipient to be selected. If this occurs, negative feedback may be sent to the sender using another built in communication structure to provide simple (good/bad) feedback remotely. Another potential problem is the likely possibility of collisions occurring between messages sent at the same time [18]. Again, a hardware structure is used to prevent this from happening. The radio transceiver will listen for a period of silence before transmitting the message, and if for some reason it is interrupted, the message will be transmitted again as soon as possible.

The remote feedback structure works by mirroring the same feedback to both the remote unit and a module in the local neural network. This module allows for verification that the correct feedback is being sent to help fix the problem. Units then provide weighted feedback to the “problem” unit, mimicking the Backpropagation algorithm which is often used for general-purpose neural systems. This feedback structure is similar to mirror neurons in the biological brain, which react to the actions of another animal as if they were being done by the observer.

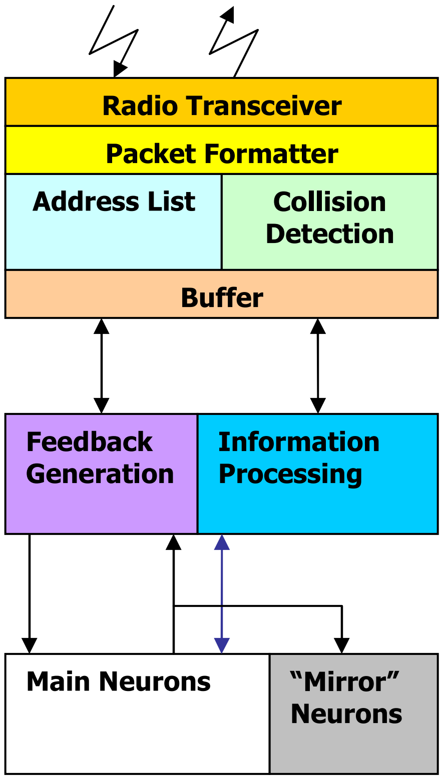

Figure 7 – Proposed communication design capable of possible language development and transmission of feedback and environment information

Much of the inter-device communication will remain elastic after implementation. There are two key reasons for this decision: it will allow the system to better adapt to its environment in terms of optimized communication, and at the same time, allow for experimentation to take place to determine if and how a form of language develops throughout the system. If this does not occur, a basic fixed-method of communication will be used to replace the proposed elastic method.

4.5 Neural Implementation

For improved neural network routing capabilities from those of fully-analog designs, as well as the benefit of FPGA configurability of the prototype design, a stochastic PWM-based neural system is designed. Stochastic PWM has an exceptional ability of imitating analog processing with digital signals. If it is presented to a resistor-capacitor filter, there is no noticeable difference between high-frequency stochastic PWM and an analog input.

4.5.1 Neuron Prototype Design

In the FPGA prototype design, each neuron will consist of a medium-resolution counter and digital comparator. After a period of time, the counter is stored and reset. The stored version of the counter is compared to the current sigmoid (tanh) and sigmoid derivative (sech^2) [15] to generate stochastic outputs based on the functions.

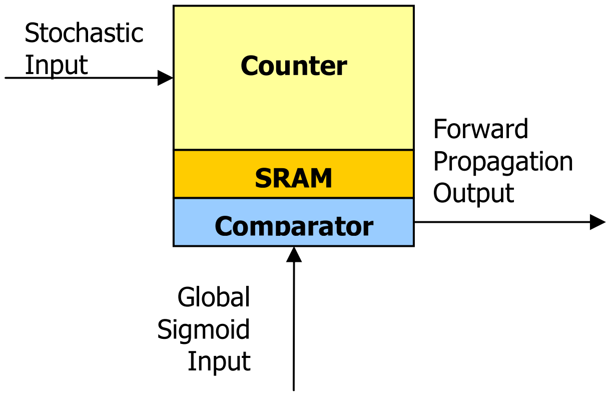

Figure 8 – Simplified neuron forwards-propagation design using stochastic inputs

4.5.2 Dendrite Tree Prototype Design

A form of a dendrite tree is necessary in this design to form a sum of the input signals. Rather than calculating the sum at each input cycle, the inputs are sampled sequentially, where each has an impact on the input of the neuron. Enabled dendrites form a shift register to multiplex the synapse outputs and form a single neuron input.

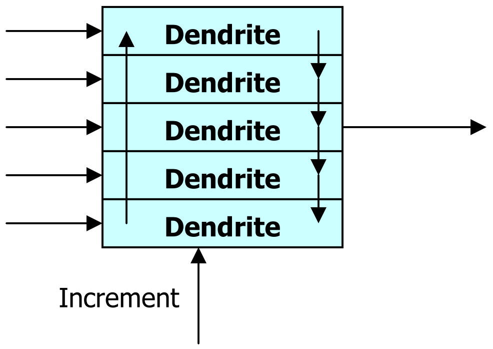

Figure 9 – Dendrite tree design utilizing a shift register for multiplexing inputs

4.5.3 Synapse Prototype Design

The basic synapse design will involve a stored weight in SRAM which is serialized with the use of stochastic global bit-enables. The serial weight is XOR'd with the input signal to provide multiplication.

Several schemes of weight update, including the popular methods of Backpropagation and Hebbian learning will be used.

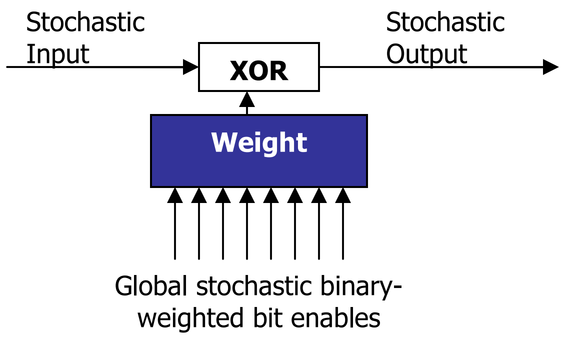

Figure 10 – Simplified forwards-propagation synapse design using local and global stochastic signals

5 Applications

5.1 Military Surveillance

Audio and video devices may be used with the miniature robots to observe, share, and store information. The units will be able to act autonomously as a group, meaning that limited human interaction is required for any task other than viewing the stored information. [13]

5.2 Mining or Agriculture

A swarm of medium-sized robots may be used for a task such as operating a mine or harvesting a field. Each robot could be equipped with tools for the specific task, and operate them either based on neural-feedback, a fixed program, or a combination thereof.

5.3 Industrial Inspection

A swarm of small robots may be used for a task such as inspecting an airplane or industrial equipment. Each robot would be equipped with high-resolution sensors, and could learn to detect any potential problems using neural network feedback.

6 References

[1] Baretto, G. et al. *A Distributed Robotic Control System Based on a Temporal Self-Organizing Neural Network.* IEEE Transactions on Systems, Man and Cybernetics, Part C. [Online]. Pp. 346-357, 2002. Available: http://ieeexplore.ieee.org/iel5/5326/26422/01176884.pdf

[2] Baker, J. et al. *CMOS - Circuit Design, Layout, and Simulation*. New York: IEEE Press, 1998.

[3] Estrada, F et al. *Local Features Tutorial*. [Online]. Available: www.cs.toronto.edu/~jepson/csc2503/tutSIFT04.pdf

[4] Haykin, S. *Neural Networks: A Comprehensive Foundation.* (2nd Ed.). New Jersey: Prentice-Hall. 1999.

[5] Hokelek, I. et al. *Dynamic Survivable Resource Pooling in FPGA-based Distributed Robotics System*. ICNSC '06. Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control. [Online]. Pp. 101-106, 2006. Available: http://ieeexplore.ieee.org/iel5/11076/35110/01673125.pdf

[6] Jackson, G. et al. *Pulse Stream VLSI Neural Systems: Into Robotics*. ISCAS '94., 1994 IEEE International Symposium on Circuits and Systems. [Online]. Pp. 375-378, 1994. Available: http://ieeexplore.ieee.org/iel2/3224/9174/00409604.pdf

[7] Janét, J. et al. *Using Control Networks for Distributed Robotic Systems*. Proceedings of the 1999 IEEE International Conference on Robotics & Automation. [Online]. Available: http://ieeexplore.ieee.org/iel5/6243/16780/00772515.pdf

[8] Kolen, J. F. and Kremer, S. C. *Dynamic Recurrent Networks.* New York: IEEE Press. 2001.

[9] Lowe, D. G. *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision. [Online]. Pp. 91-110, 2004. Available: http://cs.ubc.ca/~lowe/papers/ijcv04.pdf

[10] Mead, C. A. and Conway, L. *Introduction to VLSI Systems*. Addison-Wesley Publishing Company, Inc. 1980.

[11] Minsky, M. and Papert, S. *Perceptrons: An Introduction to Computational Geometry.* Massechusetts: MIT Press. 1969.

[12] Rao, M. A. and Srinivas, J. *Neural Networks: Algorithms and Applications.* England: Alpha Science. 2003.

[13] Rybski, P. et al. *Performance of a Distributed Robotics System Using Shared Communication Channels.* IEEE Transactions on Robotics and Automation. [Online]. Pp. 713-727, 2002. Available: http://ieeexplore.ieee.org/iel5/70/22928/01067993.pdf

[14] Soucek, B. *Neural and Intelligent Systems Integration.* New York: John Wiley and Sons. 1991.

[15] Stagg, Malcolm. *A Dynamic Analog Concurrently-Processed Adaptive Chip*. 2006.

[16] Stagg, Malcolm. *VORTECS 3D: VLSI Object Recognition Trainable Embedded CMOS System*. 2005.

[17] ST Microelectronics. *VS6650: 1 Megapixel SMIA Camera Module*. [Online]. Available: http://www.chipcatalog.com/ST/VS6650.htm

[18] Wang, J. and Premvuti, S. *Resource Sharing in Distributed Robotic Systems Based On A Wireless Medium Access Protocol (CSMNCD-W).* Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. [Online]. Pp. 784-791, 1994. Available: http://ieeexplore.ieee.org/iel2/3221/9157/00407549.pdf

[19] Wang, J. *On Sign-board Based Inter-Robot Communication in Distributed Robotic Systems*. Proceedings of the 1994 IEEE International Conference on Robotics and Automation. [Online]. Pp. 1045-1050 vol. 2, 1994. Available: http://ieeexplore.ieee.org/iel2/941/8081/00351219.pdf