# An interactive multi modal system for mobile robotic control

Akbar Ghobakhlou [1] and Remco Seesink [2]

[1]Department of information Science, University of Otago, Dunedin, New Zealand
[2]Faculty of Computer Science, University of Twente, Enschede, The Netherlands

(E-mail : [1]Akbar@infoscience.otago.ac.nz, [2]r.a.seesink@student.utwente.nl)

## Abstract

*This paper describes an interactive multi modal system for controlling a mobile robot. The robot exhibits intelligent concurrent behaviours to navigate around and avoid obstacles. It uses a broad range of different modular input and output devices. A small word recognition system based on Evolving Connectionist System is employed to recognise the voice commands. An on-line adaptation feature allows the neural network to adapt to new speakers.*

**Keywords: Speech Recognition, Evolving Neural Networks, Fuzzy Control, Robotic Control**

## 1.Introduction

For the upcoming ANNES 2001 conference we intend to build a robot to perform several different tasks.

The first task for the robot is to welcome the guests to the ANNES conference. A speech synthesizer is used to accomplish this task. Additionally, The robot will inform attendees of the conference of upcoming events during the conference while it is wandering autonomously in the atrium in the Commerce building of the university of Otago.

Secondly the robot will be used as a surveillance demonstration. The robot is controlled on a remote computer. This computer will be equipped with a word based recognition system [2] for simple voice commands. The speech recognition includes an online speaker adaptation feature which allows adaptation of a new speaker by the system A keyboard will be used for additional navigation. If the robot detects obstacles (using sonar), it can override the voice commands in order to avoid them. The computer screen will display sensory information from the video as well as the sonar of the robot.

Finally the robot will introduce some of the other demonstrations. To do this it will travel to the location of the demonstrations. A short description of the demonstration will be given by the robot via a voice command.

## 2.Robotic Control

### 2.1.Hardware Specification

To accomplish these tasks the robot is equipped with various hardware devices. The robot platform is a Pioneer 2 DX from ActivMedia [9]. It has a front sonar array with eight sonar devices placed in a half circle. It has two wheels that are independently powered and a castor wheel for dead reckoning.

The robot hardware is controlled by the P2OS (Pioneer 2 Operating System) onboard, which runs on a siemens 88C166-based microcontroller. A connection is maintained with an external computer using a serial radio modem.

On top of the robot is a Sony D30/D31 pan-tilt-zoom camera mounted. The pan-tilt-zoom function is controlled by commands sent through the radio modem connection. Figure 1 shows a robot with this configuration.

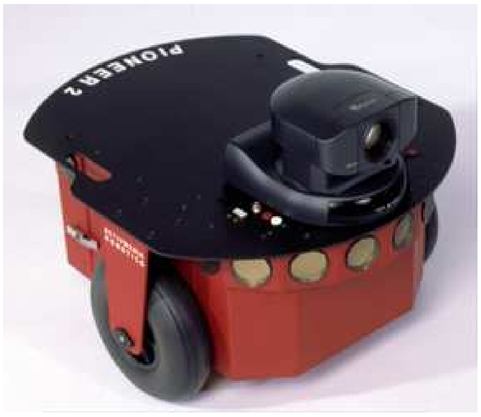

Figure 1: A pioneer 2 DX robot with camera.

The video signal is broadcast with a 2.4 Giga hertz transmitter on top of the robot to a receiver on the remote computer. The receiver is connected to a PV-BR878P frame grabber.

To give the robot speaking capabilities we equipped it with computer speakers hooked up to a radio receiver. A radio transmitter was constructed from a soldering kit.

The person controlling the robot uses a microphone to give voice commands to the robot.

### 2.2.Fuzzy control mechanism with Saphira

The robot from ActivMedia comes with Saphira [7] software. Saphira is a robotic control software layer, which abstracts platform specific robotic control functions. Saphira has been used for several different robot platforms.

Programming a robot can be a difficult task. Saphira makes this task easier because it lets you define multiple concurrent behaviours. The behaviours are executed using a round robin algorithm to avoid expensive OS solutions such as threads. Behaviours should remain computationally simple. This mechanism provides a way to decompose the problem efficiently into several sub problems. The different behaviours produce control signals, which are blended together. This unified control signal is then defuzzyfied and applied to the robot. This way a divide and conquer strategy can be implemented using behaviours. Behaviours can be started, stopped, suspended and activated.

### 2.3.Architecture

All robotic implementations attempt to translate its internal representation and sensory information to a set of control commands for the robot. We have given this robot a broad range of different sensory inputs, as well as control and output sources. This is summarized in Figure 2.

The upper layer of the diagram corresponds to the inputs of the robots. The next layer corresponds to all pre-processing tasks. These tasks convert the raw inputs to a more manageable form when needed. The decision making layer consists of all the concurrent behaviours that ultimately control the robot. The post-processing layer performs any complex calculations necessary before outputs can be generated. Finally the output layer represents all the items that receive output from the whole system. The pre-processing and post-processing layers enable the concurrent set of behaviours to remain lightweight and very responsive.

A voice command is recorded using a headset microphone. This speech signal is processed with a speech recognizer which uses the Evolving Connectionist System (ECoS) paradigm [4]. Once the command is recognised it will be executed by a behaviour. The sonar and status information is send back from the robot to the PC. This information is integrated within Saphira. Status information includes the battery voltage level, whether or not the motors are stalled and the dead reckoning from the castor wheel. Saphira will use this information to update its internal map.

Currently the video stream is displayed at the screen for surveillance purposes. The pan-tilt-zoom function of the camera can be exercised by manual controls. As shown in Figure 2 future research will focus on video processing.

State information refers to the information that is used in the behaviours, and does not come directly from the sensors. All behaviours have access to state information, using a blackboard approach. This means that all behaviours have access to a shared pool of memory. Examples of state information could be the fuzzy control variables, the angle of the robot's position, the zoom value of the camera or a desired travel destination on the map.

The speech synthesizer is from the Microsoft Speech SDK. After initialisation it can be given text, which the speech synthesizer transforms into speech. This signal is transmitted to the robot using the radio transmitter connected to the soundcard. The signal is picked up by the radio receiver on the robot. We use it to give the user feedback about spoken commands, to introduce the demonstrations and welcome the guests.

Motor control is abstracted in Saphira to velocity and angle. It is also possible to use direct motor commands. The camera has a panning range of 190 degrees and a tilting range of 40 degrees. It can zoom up to a factor of twelve.

## 3.Word Recognition

The following sections describe the neural network paradigm used to performing the speech recognition task.

### 3.1.The ECoS Paradigm

The Evolving Connectionist System (ECoS) paradigm was developed to address several of the perceived disadvantages of traditional connectionist systems. It is a structurally evolving paradigm that modifies the structure of the network as training examples are presented. Although the seminal ECoS architecture was the Evolving Fuzzy Neural Network (EFuNN) [6], several other architectures have been developed that utilise the ECoS paradigm. These include the minimalist Simple Evolving Connectionist System (SECoS) and the Evolving Self Organised Map ESOM [1]. As stated in [5] the main principles of EcoS are:

1. ECoS learn fast through one-pass training;
2. ECoS evolve incrementally and adapt itself in an on-line manner;
3. ECoS are memory-based and memorise data exemplars from the data stream;
4. ECoS learn and improve through active interaction with other systems and with the environment in a multi-modular hierarchical fashion;

The following describes the above mentioned principles:

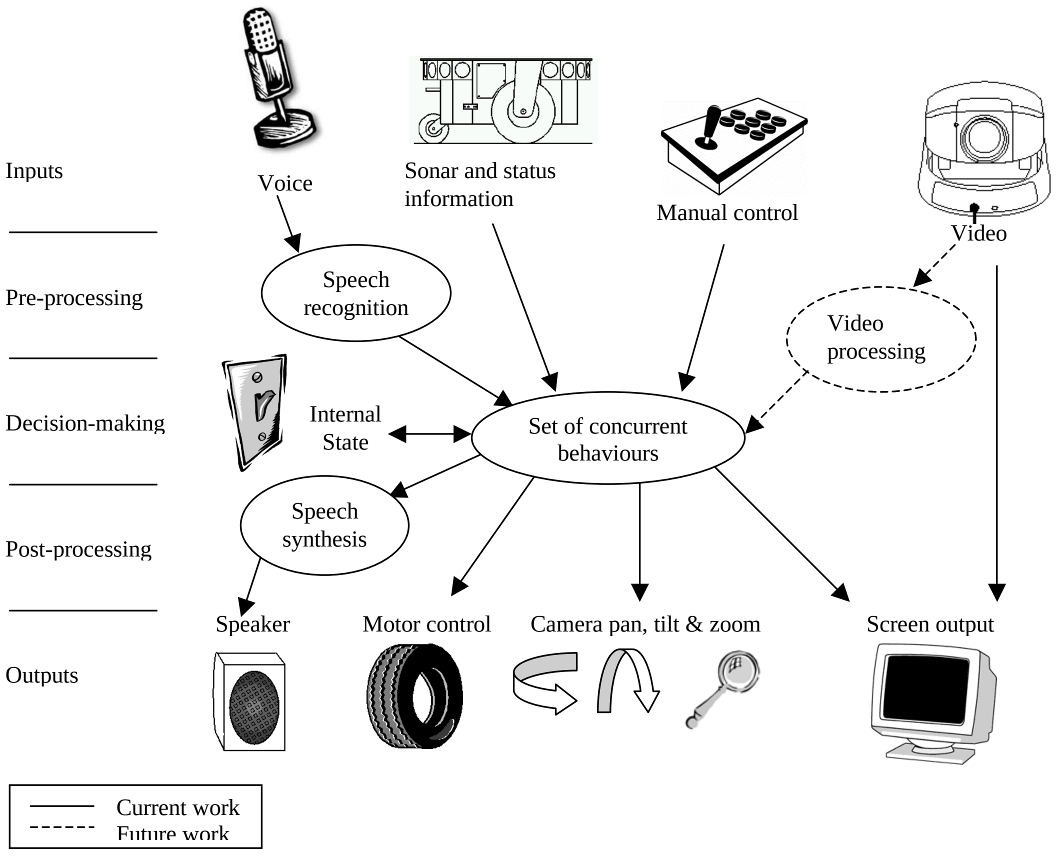

Figure 2: Control scheme for the robot.

1. The ECoS training algorithm is designed so that there is no need for an iterative training process. Through this one-pass training the maximum knowledge from the training data can be captured.

2. ECOS are intended to be used in an online learning applications. This means that new data will constantly and continuously be coming in to the system, and that it must be learned by the network without forgetting the previously learned data.

3. Initially new nodes are added to ECoS by storing some of the training examples, within the structure of the network. These examples are then either modified by exposure to further examples. Otherwise they remain the same and depending on the training parameters used they can be retrieved later.

4. ECoS networks are intended to be used in concert with other networks and systems. Thus, the learning algorithm and architecture allow for the influence of external forces, with some modules having greater importance than others.

The advantages of ECoS are that they avoid several problems associated with traditional connectionist structures. They are hard to over-train, they learn quickly, and they are far more resistant to catastrophic forgetting than most of the other models. Conversely, these advantages could cause some of ECoS disadvantages. Since they deal with new examples by adding nodes to their structure, they rapidly increase in size and can become unwieldy if no aggregation or pruning operations are applied [6]. They also have some sensitivity to their parameters, which require constant adjustment for optimum performance.

### 3.2.The Simple Evolving Connectionist System

The Simple Evolving Connectionist System (SECoS) is a the simplest implementation of the ECoS paradigm [10, 3]. SECoS are a simpler version to EFuNN which make them easier to understand and analyse.

### 3.3.The SECoS Architecture

Figure 3 is a simplified graphical representation of the SECoS architecture. A SECoS consists of only three layers of neurons, the input layer, with linear transfer functions, an evolving layer based upon the rule layer of the EFuNN model, and an output layer with a simple saturated linear activation function . The evolving layer activation is calculated as with the EFuNN, with the exception of the distance measure $D_n$ being calculated as the normalised Hamming distance, as shown in Equation 1:

$$D_n = \frac{\sum_{i}^{I} \mid E_i - W_i \mid}{\sum_{i}^{I} \mid E_i + W_i \mid} \qquad (1)$$

where:
$I$ is the number of input nodes in the SECoS,
$E$ is the input vector,
$W$ is the input to evolving layer weight matrix.

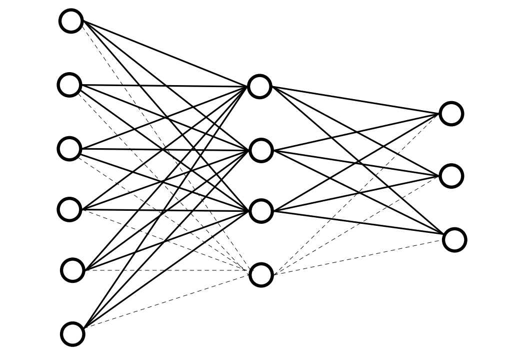

Figure 3: A simplified and exemplified diagram of a SECoS network. The dotted lines represent the creation of a new node in the evolving layer.

### 3.4.The SECoS Aggregation Algorithm

Aggregation of nodes in the evolving layer can be employed to control the size of the evolving layer during the learning process. The principle of aggregation is to merge those nodes which are spatially close to each other. Aggregation can be applied for every (or after every $n$) training examples. It will generally improve the generalisation capability of SECoS. The aggregation algorithm is as follows:

FOR each rule node $r_j, j = 1 : n$,

where $n$ is the number of nodes in the evolving layer and $W1$ is the connection weights matrix between the input and evolving layers and $W2$ is the connection weights matrix between the evolving and output layers.

- find a subset $R$ of nodes in evolving layer for which the normalised Euclidean distances $D(W1_{r_j}, W1_{r_a})$ and $D(W2_{r_j}, W2_{r_a})$ $r_j, r_a \in R$ are below the thresholds $wThr$.
- merge all the nodes from the subset $R$ into a new node $r_{new}$ and update $W1_{r_{new}}$ and $W2_{r_{new}}$ using the following formulae:

$$W1_{r_{new}} = \frac{\sum_{r_a \in R}(W1_{r_a})}{m} \qquad (2)$$

$$W2_{r_{new}} = \frac{\sum_{r_a \in R}(W2_{r_a})}{m} \qquad (3)$$

  where $m$ denotes the number of nodes in the subset $R$.

- delete the nodes $r_a \in R$

### 4.Experimental Procedure and Results

The motivation behind designing this experiment was to examine the performance of SECoS on isolated spoken word recognition. There were seven words used as commands to control the robot. Four commands, 'left' 'right', 'go' and 'stop' are used for navigation. The words 'one', 'two' and 'three' trigger the robot to give a spoken description of the corresponding demonstrations.

The speech data was recorded in a quiet environment to obtain clean speech signals. The recording was performed using a headset microphone. The speech was sampled at 22.05 kHz and quantised to a 16 bit signed number. Each word was uttered 5 times with distinct pauses in between. The words were then carefully manually segmented and labelled.

The speech data used in the experiments were obtained from two groups of speakers; group A (7 male and 7 female) and group B (2 male and 2 female). The speech data from each group were divided into two sets; training set A, testing set A, training set B and testing set B.

Training set A with $294$ examples was obtained from 3 utterances of each word. The remaining 2 utterances were used in the testing set A for a total of $196$ examples. Training set B with $112$ examples was obtained from $4$ utterances of each word in group B. Testing set B with $28$ examples was obtained from the remaining $1$ utterance of each word in group B.

Spectral analysis of the speech signal was performed over 20 msec with Hamming window and 50% overlap, in order to extract Mel Scaled Cepstrum Coefficients (MSCC) as acoustic features. In order to overcome the variabilities in the size of the MSCC obtained from each word segment, Discrete Cosine Transformation (DCT) was applied in the following manner.

For an $m$ frame segment, DCT transformation will result in a set of $m$ DCT coefficients for each feature. This sequence is truncated to achieve a fixed-size input vector consisting of $20 \times d$, where $d$ is the dimensionality of the feature space. Since five MSCCs were taken as acoustic features, the DCT transformation was applied over each acoustic observation. Therefore, the first twenty coefficients of each acoustic observation were retained to make a fixed-size of 100 inputs for every word segment.

### 4.1.Training SECoS

A SECoS was initialised with 100 nodes in the input layer and 7 nodes in the output layer (one for each word). In phase one of the experiments the SECoS was trained on the training set A with the parameters shown in Table 1. Aggregation was used to control the size of the evolving layer during the learning process.

Table 1: SECoS's Training and Aggregation Parameters

| | |
|---|---|
| Error Threshold | 0.01 |
| Aggregation Threshold | 0.18 |
| Number of Examples Before Aggregation | 50 |

There were 156 nodes created during the training period. The trained SECoS was then tested on training set A and both testing sets A and B. The SECoS performance on training set A was 100%. Table 2 shows the performance of the SECoS on testing sets A and B.

Table 2: Performance of SECoS on Speech Commands

| | Testing Set A | | Testing Set B | |
|---|---|---|---|---|
| Words | Positive Accuracy | Negative Accuracy | Positive Accuracy | Negative Accuracy |
| Go | 94.12 | 98.69 | 83.33 | 98.25 |
| Left | 88.24 | 100.00 | 91.67 | 100.00 |
| Right | 88.24 | 100.00 | 83.33 | 100.00 |
| Stop | 100.00 | 100.00 | 100.00 | 100.00 |
| One | 97.06 | 98.53 | 100.00 | 97.00 |
| Two | 94.12 | 100.00 | 96.15 | 99.00 |
| Three | 100.00 | 97.06 | 92.31 | 98.00 |
| Overall | 95.29 | 99.29 | 98.94 | 98.94 |

To improve the performance over the testing set B (ie: new speakers), the trained SECoS was additionally trained on training set B. 11 additional nodes were added to the new SECoS to accommodate the variations in the new speakers. The performance of the adapted SECoS on its training set B was 100%. The adapted SECoS was then tested on the original training (training set A) and testing sets A and B. Performance of the adapted SECoS on training set A was remained unchanged. Table 3 illustrates the performance of the SECoS on testing sets A and B after adaptation on new speakers.

Table 3: Performance of SECoS on Speech Commands After Adaptation

| | Testing Set A | | Testing Set B | |
|---|---|---|---|---|
| Words | Positive Accuracy | Negative Accuracy | Positive Accuracy | Negative Accuracy |
| Go | 100.00 | 97.39 | 100.00 | 100.00 |
| Left | 88.24 | 100.00 | 100.00 | 100.00 |
| Right | 88.24 | 100.00 | 100.00 | 100.00 |
| Stop | 100.00 | 100.00 | 100.00 | 100.00 |
| One | 97.06 | 99.26 | 100.00 | 100.00 |
| Two | 100.00 | 98.53 | 100.00 | 100.00 |
| Three | 97.06 | 100.00 | 100.00 | 100.00 |
| Overall | 95.80 | 99.31 | 100.00 | 100.00 |

## 5.Discussion

The results illustrate that SECoS is quite capable of memorising an entire training set. When SECoS initially trained on training set A or after further trained on training set B (data from new speakers), the network performed with 100% accuracy over both training sets A and B. This memorisation is despite the fact that there are very much fewer neurons in the evolving layer than there are examples in the training sets.

Despite the very high level of memorisation shown by the network, a high level of generalisation was also demonstrated. Only two words 'left' and 'right' from testing set A was recognised with an accuracy of 88.24%, while just two words from testing set B, 'go' and 'right' were recognised with a positive accuracy of 83.33%. For examples from all (both known and unknown) speakers (testing sets A and B), the true negative accuracy of the network is consistently over 97%. Therefore, the SECoS is able to both memorise training data as well as generalise to new examples and new speakers to a very high degree.

Despite the very small size of the training set B adaptation of the SECoS on training set B greatly improved the performance of the network over testing set B, with positive and negative accuracies of 100% being recorded for each word. The network also memorised training set B perfectly, with 100% recognition of all words. Despite this high level of adaptation, the accuracy over testing set A did not decline.

Few nodes were aggregated in relation to the total size of the network. This is because of the very high aggregation threshold used during training, which caused very few nodes to be aggregated.

Although the milled aggregation has improved the generalisation capability of the network, it highlights a problem with aggregation during training process.

## 6.Implementation and user interface

This system is implemented using Microsoft Visual C for the framework and the user interface. The user interface shows all the different sensory information from the robot.

The Saphira window displays robot status information and sonar readings. When a map is provided, this will be integrated in the sonar reading screen.

The user interface allows the user to control the robot either manually, by voice commands, or autonomysly. The manual controls include manipulation of the camera and motor control. Parameters such as speed and turning angle can be adjusted. There are other basic controls provide such as connecting to the robot and switching sonar on and off. The speech synthesizer can be controlled from this window. It is possible to enter text for the robot to speak complementary to the robot built-in feedback speech. Video will be displayed on the screen.

The user interface includes controls for the speech recogniser. A record button will record the voice command when pressed. The audio signal will then be passed to the speech recogniser and produce a command. This command is displayed on the screen and passed to a behaviour. The behaviour will use the speech synthesizer to give feedback on the command. On the control window is an adapt button, which will allow the speech recogniser to adapt online to a new speaker.

The behaviours we implemented are listed below. These behaviours are ultimately responsible for controlling the robot.

- Obstacle avoidance: This will try to prevent collision with obstacles in front of the robot. Note that there is no direct sensory information for the back of the robot, so the safest thing to do is not to drive backwards. This behaviour is used for autonomous navigation.

- Constant velocity: This will cause the robot to keep on moving after other behaviours have told it to slow down. It is complementary with the obstacle avoidance for the autonomous navigation.

- Speech interpreter: This behaviour will apply speech commands to the robot's control. This behaviour will check if there are object in conflict with the commands and produce feedback if commands can not be executed.

- Manual controller: This will interpret manual controls from the user. It can reduce sensitivity for obstacle avoidance, and suspend the constant velocity behaviour. This behaviour is intended for the surveillance task and to control the robot in a fast and efficient way if needed.

- Greeting guests: This behaviour will cause the robot to greet the guests at the conference and inform them about the program of the conference and upcoming events.

## 7.Conclusions and Future research

This paper illustrates that it is feasible to control a mobile robot using various modules. This is done by using a divide and conquer approach. By implementing multiple concurrent behaviours it is possible to have a broad ranges of different inputs to control the robot. It was also shown that this approach can generate a multitude of output results.

In addition this paper described the structure of the SECoS as an evolving network using the ECoS paradigm . The experiments carried out with SECoS for isolated word recognition to explore the performance of SECoS locally learning algorithm. It was clearly demonstrated that SECoS is capable of learning new data (from new speakers) and also shown that SECoS maintained its previously learned knowledge after adaptation on new speakers.

It was noted that applying a single aggregation threshold to the to nodes that represent different classes results in uneven aggregation. This could cause harsh aggregation of the nodes of a particular class. Further research is needed to investigated the issue of aggregation.

In future we want to extend this set of functionalities with video processing capabilities as is indicated in Figure 2 . This requires providing an interface from our software to the frame grabber.

The video processing part of this extension could be twofold. Firstly, it is possible to construct a visual sonar. In [8] is a example of this method described as it is used in the polly-system.

Visual sonar imposes some restrictions on the environment to work properly. The first of these is the back-ground-texture-constraint. This implies that the floor needs to have texture free uniform colour and that all object have other colours. A second restriction is the ground-plane constraint. This constraint guarantees that all object stand on the floor.

The visual sonar method first calculates all the pixels associated with the floor. After this process a depth chart can be calculated by looking at height of the objects on the picture.

Secondly, an extension could use landmark detection by exploiting colour or pattern recognition. This information can be integrated into the internal mapping software of saphira to update the position of the robot on the map. Saphira already uses corridors and doors to estimate its position, but in real life environments this is often unreliable.

These video-processing algorithms can drive new behaviours for controlling the robot. These behaviours can also actively use the pan-tilt-zoom function of the camera to broaden their range and potential application.

## Acknowledgments

We would like to thank professor N. Kasabov for his kind supervision. We would also like to thank the KEL laboratory, the technical staff, Mike Watts and Qun Song.

## References

[1] Da Deng and Nikola Kasabov. ESOM: An algorithm to evolve self-organizing maps from on-line data streams. In *International Joint Conference on Neural Networks (IJCNN'2000), Como, Italy*, 2000.

[2] A. Ghobakhlou, Q. Song, and N. Kasabov. Rokel: the interactively learning and navigating robot of the knowledge engineering laboratory at otago. *ICONIP/ANZIIS/ANNES'99 Workshop, Dunedin, New Zealand, November 22-24, 57-59*, 1999.

[3] A. Ghobakhlou, M. Watts, and N. Kasabov. On-line expansion of output space in evolving fuzzy neural networks. *ICONIP 2000 7th International Conference on Neural Information Processing, Taejon, Korea, November 14-18, 891-896*, 2000.

[4] N. Kasabov. The ECOS framework and the ECO learning method for evolving connectionist systems. *Journal of Advanced Computational Intelligence*, 2(6):195–202, 1998.

[5] N. Kasabov. The ECOS framework and the ECO learning method for evolving connectionist systems. *Journal of Advanced Computational Intelligence*, 2(6):195–202, 1998.

[6] Nikola Kasabov. Evolving fuzzy neural networks - algorithms, applications and biological motivation. In Takeshi Yamakawa and Gen Matsumoto, editors, *Methodologies for the Conception, Design and Application of Soft Computing*, volume 1, pages 271–274. World Scientific, 1998.

[7] K. Konolige and K. Myers. The saphira architecture for autonomous mobile robots. *http://www.ai.sri.com/ konolige/saphira/*, 1996.

[8] D. Kortekamp, R.P. Bonasso, and R. Murphy. Artificial intelligence and mobile robots case studies of successful robot systems. pages 125–139. The MIT Press/American Association for Artificial Intelligence, 1998.

[9] ActivMedia Robots Web Site. Software, documentation and technical support. *http://robots.activmedia.com*.

[10] Michael Watts and Nik Kasabov. Simple evolving connectionist systems and experiments on isolated phoneme recognition. In *Proceedings of the first IEEE conference on evolutionary computation and neural networks, San Antonio, May 2000*, pages 232–239. IEEE Press, 2000.