Incorporating Reactive Learning Behaviour into a Mini-robot Platform

Mark Elshaw
and
Debra Lewis
and
Stefan Wermter

School of Computing and Technology, Centre for Hybrid Intelligent Systems, University of Sunderland, St Peter's Way, Sunderland SR6 0DD, United Kingdom
[Mark.Elshaw][Stefan.Wermter]@sunderland.ac.uk
www.his.sunderland.ac.uk

Abstract

In the Neuro-robotics Lab of the Centre for Hybrid Intelligent Systems we have organised various challenging undergraduate student projects using Mindstorm, Khepera and PeopleBot robots. In this paper we will describe a particularly interesting undergraduate student project involving the introduction of intelligent behaviour onto a mini-robot platform. In the past robots were mainly hard-coded to perform actions that limited their adaptability to their environment. As a response to this problem there has been considerable interest in producing robots that can learn. By developing a fly-catcher robot scenario using a mini-robot platform it was possible to consider the incorporation of learned intelligent behaviour for navigation and the use of object recognition. In doing so neural network learning was incorporated into a mini-robot. This work offers an insight for robot uses into issues associated with intelligent learning on mini-robots that are also applicable to those using more sophisticated robots.

1. Introduction

The Neuro-robotics Lab in the Centre for Hybrid Intelligent Systems at the University of Sunderland has a wide selection of sophisticated intelligent robots within its purpose built laboratory including two large PeopleBot robots, mindstorms, Kephera etc. One important component of our work is to organise challenging undergraduate student projects using these robots. This has proved very successful in the past with projects on the association of language with robot actions on the PeopleBot, navigation on an interactive Khepera, neural network based object manipulations using the Khepera, vision processing on the PeopleBot for landmark navigation and language recognition and creation on a PeopleBot tour guide robot (http://www.his.sunderland.ac.uk/Robots_frame.html). In this paper we will describe a particularly interesting undergraduate student project by Debra Lewis using a mindstorm robot. In nature fly-catching animals may have restricted computational resources and this motivated our interest to try to model a fly-catcher as a small mini-robot. These insights include real-time processing, efficient programming techniques, multi-modal inputs, image processing and incorporating learned behaviour.

This paper considers the introduction of intelligent learned behaviour on a mini-robot platform to perform the functions of navigation and recognition. This system combines readings from the two modalities vision and touch to navigate around an environment, performs object recognition and captures an object in real-time. Despite the very restrictive nature of a mini-robot such as the mindstorm robot that was used in this case intelligent learning behaviour was incorporated into the system. This involved the introduction of a neural network to perform navigation and the use of object recognition. Although there is considerable research into mini-robots, learning has previously had little impact.

The mindstorm robot platform is very basic. It contains a RCX Microcomputer, LEGO elements, motors, light and touch sensors, and an infrared transmitter [1]. The RCX is an autonomous microprocessor that is programmable using the computer language Not Quite C [2]. Once programmed the RCX takes simple inputs from the environment based on sensor readings, processes the data and then produces the appropriate output by typically manipulating the motor directions [3, 4, 5]. Although these robots are restricted in the number of sensors that can be put on the robot and the memory space (6k) to download programs, they are suitable for modelling simple intelligent learning behaviour to perform actions autonomously. The Not Quite C programming language is based on 'C', but it is very limited in comparison. It is only possible to use 32 global and 16 local integer variables. Hence, Not Quite C is not able to provide the complex algorithms and real-valued functions that are typically associated with machine learning.

The add-on accessory set for vision contains a basic digital camera and basic software for object recognition. The Vision Command Set offers possibilities to have the robot perform actions based on what it sees. However, this set does lack sophisticated support algorithms typically required for object recognition. Further, although the mindstorm robot software and the Vision Command software are designed to work together, they are clearly distinct with the mindstorm software working autonomously on the robot and Vision Command through the PC.

In order to examine the suitability for including learned behaviour on a restricted mini-robot platform a fly-catcher scenario was devised. The scenario involved putting the robot into an environment and having it look for an object, known as the fly. If the robot could not see its prey it moved around until it achieved line-of-sight. Once the fly was observed the fly-catcher was required to go to it, grab it and then relocate the fly. This paper includes an examination of intelligent robot systems, a description of the intelligent navigation and object recognition approaches used on the fly-catcher robot and finally a discussion of the fly-catcher robot's overall performance.

2. Intelligent Robot Systems

In the past robots such as the Honda robot required hard-coding of all behaviour and so could not learn and adapt to changes in their environment [6]. To overcome the need for hard-coded robots researchers are examining intelligent learning in robots. Robot learning according to Araújo and Barreto (2001) [7] is a challenging domain due to its complexity, restrictions on the amount of training data available and the real-time decision-making. Below is an examination of a few examples of robotic learning systems that have been developed.

For instance, in our research [8, 9] using the MIRA[1] PeopleBot robot as part of the MirrorBot project we developed a modular self-organising model that controls robot actions using language instruction. The MirrorBot project examines perceptual processes using models of cortical assemblies and mirror neurons to explore the emergence of semantic representations of actions and concepts in a neural robot. In this context we focused on how language instructions for actions were modelled in a self-organising memory. In particular it focused on the neurocognitive clustering of actions based on the part of the body that performs the action and regional modularity for language areas in the brain. This approach used actual sensor readings from the robot to represent low level semantic features of the actions as the input to the neural network and also as the basis for the robot's behaviour.

Furthermore, Calabretta et al. (1998) [10] examined an intelligent approach for control of a robot to perform litter collection. This approach broke down behaviour into sub-elements that match diverse neural modules as an implementation of evolutionary adaptive procedures. Three different architectures were considered for litter collection: a feedforward network, a hardwired modular architecture which allowed the required behaviour to be controlled by different neural modules, and finally the duplication-based modular architecture where the modules were not hardwired but added during the evolutionary process. It was found that the architectures with modules outperformed those with a basic network structure. For the hardwired modular architecture the evolved individuals always developed a single module to control the left motor, the pick-up process and used two competing neural modules for the right motor. For the duplication-based modular approach the evolved individuals used both neural modules to control the left motor, the right motor, the pick-up procedure and the release process.

Kazer and Sharkey (2001) [11] developed a model of how the hippocampus combines memory and anxiety to produce novelty detection in a robot. In the network structure layers CA3 and CA1 depicted the same regions in the hippocampus. The network weights linked these layers by performing Hebbian learning. Categorising input vectors as novel or familiar identified the amount of anxiety. The novelty/familiarity categorisation relied on activation, which was dependant on inhibition. The model was found to offer a direct association

---

[1] MIRA is the robot used in the MirrorBot Project and MIRA stands for MIrror neuron Robot Agent.

between anxiety and Hebbian-learning models of hippocampal learning.

A learning robot was devised by Pérez-Uribe (2001) [12] that used a trial-and-error learning approach. The computerised systems used the temporal-difference approach to learn to predict by using reinforcement learning. Pérez-Uribe (2001) [12] used a neural model in the learning robot that decided between three possible actions: perceive a pattern while moving left; perceive a pattern while moving right; and perceive no pattern. Once the correct selection was made the operator pushing a button gave rewards. Such an approach is of interest as it offers an opportunity for robot learning through human teaching.

Recently it was discovered that mirror neurons located in the rostral region of primates fired not only due to performing an action but also observing it [13, 14]. This finding could have a significant impact on learning robots as it offers an approach for robot learning through imitation and multi-modal information fusion. Based on this finding Demiris (2002) [15] devised an architecture to achieve robot learning through imitation using behaviour and forward models. The behaviour model was given information on the current state and the goal and produced the required motor commands. The forward model then created the expected next state based on the output from the behaviour model. The predicted state was compared with the actual state of the demonstrator to produce an error signal. The error signal was used to create a confidence value to establish the confidence by which a particular behaviour was identified. The approach used two simulated robots, the demonstrator and the imitator. The demonstrator robot was observed by the imitator robot performing a single action or a series of actions and then required to predict what was being performed from a stored set of actions or action orders. As the demonstrator performed the action or series of actions the confidence in certain actions or series of actions reduced as it became less likely that they were being performed and the confidence in the final prediction increased. This therefore gave the robot the ability to imitate the demonstrators and understand what was being performed.

3. Reactive Fly-Catcher Mini-robot System

In order to perform the actions outlined in our scenario the robot received binary inputs from two touch sensors on the front of the robot and one at the back. Using readings produced by these sensors the conditions of the motors were altered based on learned behaviour. Furthermore the fly-catcher used object recognition to locate the fly and capture it. A photograph of the fly-catcher robot and the fly prey is given in Figure 1.

3.1 Neural Network for Intelligent Navigation Behaviour

It was decided to use a neural network for navigation control on the fly-catcher robot as this learning technique has proved successful for navigation in more sophisticated robots and it gives us a chance to consider the viability of the technique for use on a mini-robot. Given the restrictions imposed by the mindstorm robot platform the selection of the most appropriate neural network to perform this task in an autonomous manner was critical. Two basic architectures were considered the self-organising map (SOM) and the back-propagation multi-layer perception (MLP).

The SOM consists of two layers, an input layer and output layer, and is an unsupervised training approach [16]. Such a network learns by creating a topological representation of the critical characteristics of the input through a pattern of active and inactive units. Although having a SOM on an autonomous robot would enable it to investigate its environment it is likely that this would require a connected PC for a mini-robot to perform the computations required and so the mini-robot would no longer be autonomous.

It was decided to produce the intelligent navigation behaviour using the supervised learning approach of the multi-layer perceptron (MLP). A supervised approach involves training the network using both the inputs and the required outputs. McClelland and Rumelhart (1986) [17] pioneered the MLP, which combines processing neurons into at least three layers, the input layer, the middle hidden layer and output layer. Figure 2 provides a typical representation of a MLP network.

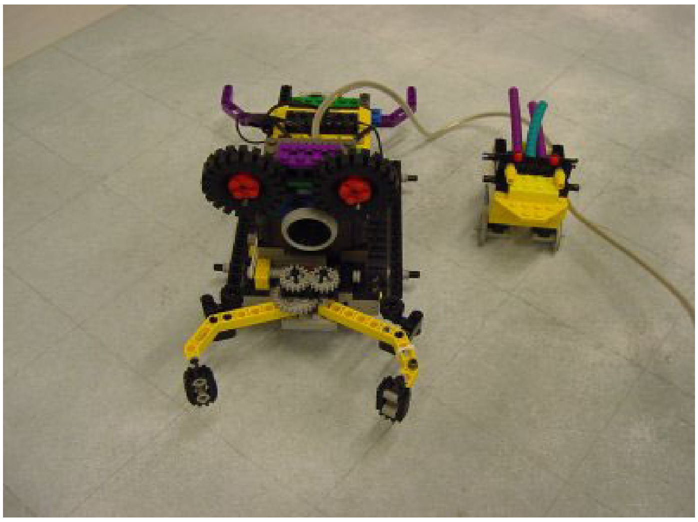

Figure 1. The fly-catcher robot and its fly prey.

The learning rule typically used for the multi-layer neural network is the back-propagation rule that allows the network to learn to classify. This rule creates the output of the network compares this with the required output and by propagating the error back through the network alters the weights to reduce the error [18].

The connections between sensors (three inputs) and motors (two outputs) and the two neurons in the hidden layer of the navigation MLP are shown in Figure 3. It can be seen from Table 1 that the input units received values from the touch sensors to indicate if the robot is blocked (1 if blocked, 0 if not blocked). The robot learned the direction that the motors need to take to avoid the obstacle, which is represented by +1, 0 or –1 (+1 forward, 0 turn off motor and –1 move back). For instance, when the left front sensor was pressed, the required action would turn the robot to the right by turning off the left motor and have the right motor reverse. This would have the output representation of 0, -1. The network was trained using the 8 possible input and output combinations in Table 2.

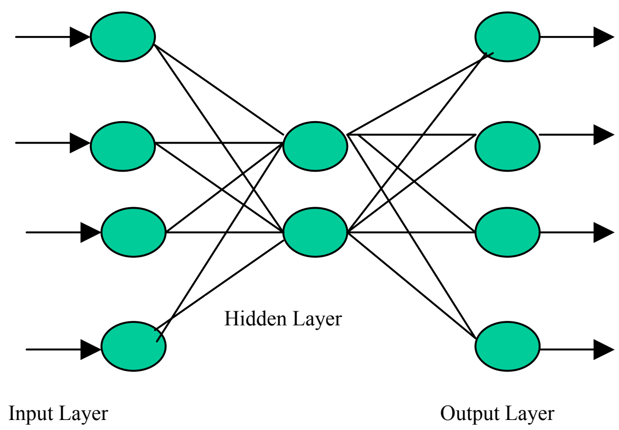

Figure 2. The multi-layer perceptron network architecture.

Input			**Output**	
Sensor Front left	**Sensor Back**	**Sensor Front right**	**Motor Left**	**Motor Right**
1	0	0	0	-1
0	1	0	1	1
0	0	1	-1	0

Table 1. Input representation for intelligent navigation.

Input			**Output**	
Sensor Front left	**Sensor Back**	**Sensor Front right**	**Motor Left**	**Motor Right**
1	0	0	0	-1
0	1	0	1	1
0	0	1	-1	0
1	0	1	-1	-1
1	1	0	0	1
0	1	1	1	0
1	1	1	0	0
0	0	0	1	1

Table 2. The 8 possible inputs and output combinations for intelligent navigation.

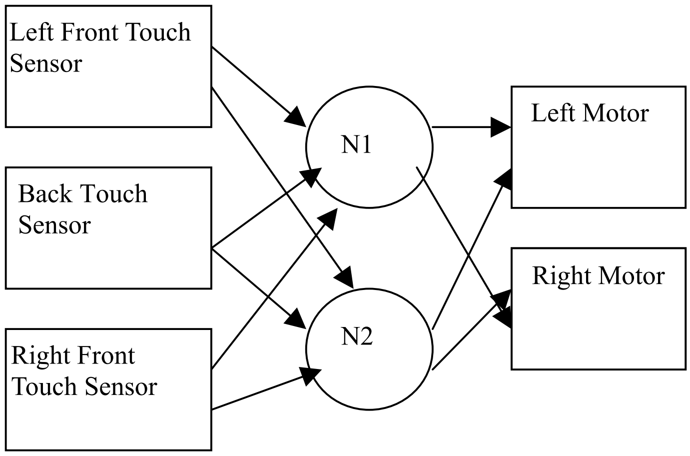

Figure 3. The Navigation MLP network structure.

3.2 Object Recognition using a Mini-robot

Intelligent robot vision and object recognition is an area of active research, which has allowed some new and novel approaches for analysing what is seen by the robot camera. A vision processing based robot by Nehmzow (1999) [19] used a SOM to cluster vision data from a camera in an autonomous manner to differentiate between images that included boxes and those that did not. By using the SOM to process the robot's sensory signals, distinct sensory perceptions were mapped onto clear areas of the network, with close perceptual patterns clustering together in an area. Of the camera images that contained boxes, 70% were correctly identified as containing boxes and of those without boxes 60% were classified correctly.

Furthermore, Weber and Wermter (2003) [20] in our Neuro-robotics Lab produced a vision system based on an associator neural network to localise an object within the visual field. The model was used to direct the MIRA robot so it altered the pan and tilt of its camera to centralise the object. The neural architecture used "what" and 'where' pathways. The 'what' pathway used two areas: input and a two-layer hidden area. The lower layer of the hidden area gained bottom-up connections from the input. The upper layer of the hidden layer received the output of the lower layer neurons. After it received this initial input, it updated its activations using its previous activations. Furthermore, input came from the connected area of the "where" pathway. The "where" neurons via recurrent weights were fully connected and also received inputs from the highest layer of the 'what' network.

Wilson and Mitchell (2000) [21] developed an artificial retina that detected high contrast objects in the line of vision and relayed this information to the host processor. The retina broke down the image into smaller subsections to help analyse the line of vision. Moreover, Cipolla (1995) [22] developed a robot that included ‘stereo’ vision in a robot that learned by experience by attaching two independent cameras.

Roy and Pentland 2002 [23] developed a robotic system called CELL that could learn shape and colour words that incorporated object recognition. Object recognition was achieved by taking multiple two-dimensional images of the object from different positions that collectively form the model of that object. Histograms of features were derived from the object models that represented them. Shape was determined by locating the boundary pixels of the object in the image. The use of multidimensional histograms to represent shape allowed the comparison of different objects. This enabled a comparison of the two-dimensional histograms that represent the objects from specific viewpoints. Out of the 15 two-dimensional histograms from the viewpoints the 4 closest for the two objects were taken and the difference summed.

In order to identify the fly, the fly-catcher mini-robot used the Vision Command set to perform colour recognition. By using this software to capture pictures at up to 30 frames per second on a 352 x 288 resolution camera the fly-catcher robot could recognise pre-defined colours. Once received, inputs from the camera are passed into the recognition software on the PC, the Vision Command set sends messages via the infrared tower to the RCX stating whether the desired colour was recognised. Based on this colour recognition the Not Quite C programmed RCX produced the appropriate motor commands. The mindstorm colour recognition package used a process that compared the RGB values for the predefined colour with the RGB value of the main colour of the object it encounters. The Vision Command software divided the visual field into various regions and so was able to detect where in the visual field the fly was and so used this information to move towards it.

Although the camera and recognition software added new functionality to a mindstorm robot, it does have two main limitations. By using RGB values lighting played a critical part in what colour the robot was actually looking for and whether it recognised the colour under different conditions. Running vision with the navigation approach caused the navigation and vision software to ‘fight’ for control of the motors and so produced the situation where the motors were being turned on and off continuously. To overcome the first limitation efforts were made to keep the lighting in the environment constant. Furthermore, it became necessary to allow the vision or navigation function currently in control of the motors to temporarily suspend the other until that function had finished with the motor output, then pass control back to the other by use of a semaphore system.

The fly-catcher determined whether to close its gripper on the recognised fly and relocate to a new position by using a light sensor on its front. If the fly was in the correct position the light sensor’s value passed a certain threshold and so the fly-catcher performed the appropriate behaviour.

When testing the navigation multi-layer perceptron using artificially created inputs off the robot it was found that the network did produce values close enough to the required output values that a thresholding approach could be incorporated in the robot. This thresholding approach stated that if the output was greater than +0.5

the motor moves forward, if it was below –0.5 it was reversed and between –0.5 and +0.5 the motor was turned off.

Once the trained reactive behaviour was incorporated in the fly-catcher, this robot and the fly were repeatedly placed into the environment at different locations. This was done to establish how well it would perform the task of navigation and object recognition under diverse circumstances. The environment was completely white, with a base 180cm by 120 cm and 60 cm high walls around the edge of the base. In order to test the navigation behaviour, block shaped obstacles of different colours to the fly were positioned in the environment. Despite the white colouring of the environment causing the reflection of the fly on the wall and block obstacles, the simple navigation behaviour of the robot enabled it to navigate round the environment and move into a position to recognise the fly in 80% of the test cases. Figure 4 shows the fly-catcher robot looking for its prey.

Furthermore, when the robot recognised the fly and repositioned itself to the front of the fly, it was able to take hold of the fly and move it from the capture site 100% of time. However, when the robot came at the fly from a different angle in only 50% of the cases did the robot grasp the fly and move it from the capture site. Table 3 provides a summary of these results.

Flies navigated to and recognised correctly	Percentage of recognised flies captured if approached from front	Percentage of recognised flies capture if approached from side
80%	100%	50%

Table 3. Performance of the fly-catcher robot.

Despite the many limitations associated with mini-robots like the mindstorm robot and its programming language Not Quite C, it was possible to overcome these to produce simple intelligent behaviour. We were able to incorporate learned behaviour using a multi-layer perceptron neural network and performed object recognition. By using this simple reactive behaviour the robot was able to navigate round an environment, avoid obstacles, recognise its prey, grasp it and take it to a new location.

It might be argued that it is possible to use much more sophisticated platforms that have a large amount of hard disk space and memory, and can be programmed using a powerful programming language straight away. However, they have a much longer learning curve and may not offer the ease of use of the mini-robot considered here. Although the level of intelligent behaviour possible with mini-robots is limited they do give a valuable insight and introduction into the problems of incorporating learning and reactive behaviour on robots especially at the level of student projects. As seen in this paper such a robot can draw attention to issues such as lighting conditions for object recognition, the conflicts that can occur when trying to combine inputs from different modalities, gripper manipulation and learning approaches for producing complex reactive behaviour.

4. Conclusion

In conclusion, as part of an undergraduate project learning behaviour was introduced into a mini-robot based on a fly-catcher scenario to perform navigation and object behaviour. By doing so it was possible to consider the introduction of learned behaviour on a mini-robot. Although, such a robot does offer many limitations it is still possible to achieve good performance by using a multi-layer perceptron for navigation and colour recognition to identify the robots prey. Furthermore, many of the issues brought out from examination of this type of robot are applicable to more sophisticated ones. It is our belief that some of the findings on the mindstorm have aided our research into more sophisticated robot behaviour on the larger PeopleBot robot.

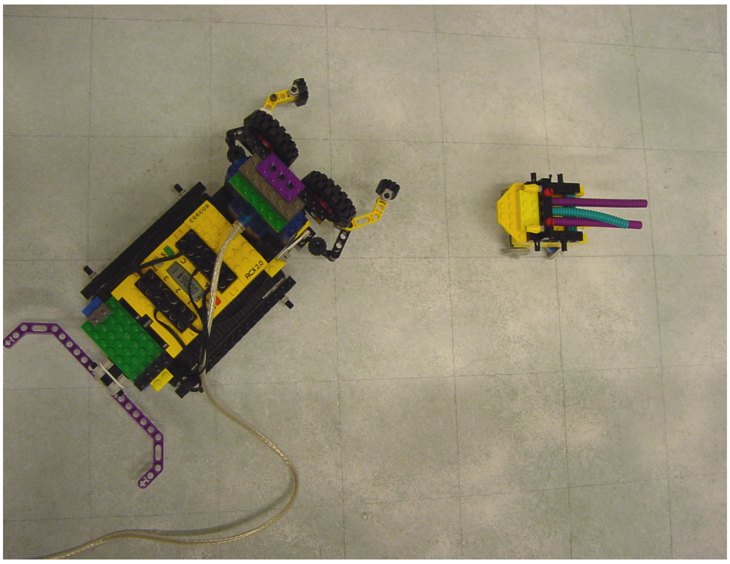

Figure 4. The fly-catcher robot looking for its prey.

Acknowledgements

We would like to thank the following members of the Centre for Hybrid Intelligent Systems for their help with this paper Sheila Garfield, John Murray and Cornelius Weber. In particular I would like to thank Harry Erwin and Christo Panchev for their assistance with the final year projects in the Neuro-robotics Lab.

References

1. Baum, D. Dave Baum's Definitive Guide to Lego Mindstorms, Apress, 2000.
2. Baum, D., Gasperi, M., Hempel, R. & Villa, L. Extreme mindstorm: advanced guide to LEGO mindstorm. Apress, 2000.
3. Nagata, J. Lego Mindstorms idea book, Starch Press, 2001.
4. Penfold, R. More Advanced Robotics with Lego Mindstorms. Bernard Babani (Publishing) Ltd, 2000.
5. Sato, J. Jim Sato's Lego mindstorms: The master's technique. Starch Press, 2001.
6. Schaal, S. Is imitation learning the route to humanoid robots. Trends in Cognitive Science, 1999; 3(6): 233-242.
7. Araújo, A. & Barrato, G. A self-organizing context-based approach to the tracking of multiple robot trajectories. International Journal of Computer Research, 2001; 10(2): 139-179.
8. Elshaw M., Wermter S. & Watt P. Self-organisation of language instruction for robot action. Proceedings of the International Joint Conference on Neural Networks. Oregon, USA, July 2003.
9. Wermter S. & Elshaw M. Learning robot actions based on self-organising language memory. Neural Networks, 2003, 16(5-6): 661-669.
10. Calabretta, R., Nolfi, S., Parisi, D. & Wagner, P. Emergence of functional modularity in robots. In Proceedings of Artificial Life VI, Los Angeles, Editors C. Adami, R. Belew, H. Kitano and C. Taylor, MIT Press, 1998.
11. Kazer, J. & Sharkey, A. The role of memory, anxiety and Hebbian learning in hippocampal function: Novel explorations in computational neuroscience and robotics. In Emergent Neural Computational Architectures based on Neuroscience, Editors S. Wermter, J. Austin, and D. Willshaw, Springer-Verlag, Heidelberg, Germany, 2001, pp. 507-521.
12. Pérez-Uribe, A. using a time-delay actor-critic neural architecture with dopamine-like reinforcement signal for learning in autonomous robots, In Emergent Neural Computational Architectures based on Neuroscience, Editors S. Wermter, J. Austin, and Willshaw, D., Springer-Verlag, Heidelberg, Germany, 2001, pp. 522-533.
13. Rizzolatti G. & Arbib, M. Language within our grasp. Trends in Neuroscience, 1998; 21: 188-194.
14. Umilta, M. Kohler, E. Gallese, V., Fogassi, L. Fadiga, L., Keysers, C. & Rizzolatti, G. I know what you are doing: A neurophysical study. Neuron, 2001; 31:31.
15. Demiris, Y. Biologically inspired robot imitation mechanisms and their application as models of mirror neurons. Proceedings of the EPSRC/BBSRC International Workshop on Biological Inspired Robotics Bristol 14-16 August, 2002.
16. Kohonen, T. Self-organizing maps. Springer Verlag, Heidelberg, Germany, 1997.
17. McClelland, J. & Rumelhart, D. Mechanisms of sentence processing: assigning roles to constituents of sentences. Parallel Distributed Processing: Explorations in the Microstructure of Cognition Vol.2, MIT Press, Cambridge MA, 1986, pp. 272-325.

18. Beale, R. & Jackson, T. Neural computing an introduction. IOP Ltd, 1990, pp. 39-128.
19. Nehmzow, U. Meaning through clustering by self-organisation of spatial and temporal information. Computation for Metaphors, Analogy and Agents, Editor C. Nehaniv, Springer-Verlag, Heidelberg, Germany, 1999, pp. 209-299.
20. Weber C. & Wermter S. Object localisation using laterally connected "What" and "Where" associator networks. Proceedings of the International Conference on Artificial Neural Networks, Istanbul, Turkey, pp. 813-820, 2003.
21. Wilson, J. & Mitchell, R. Object detecting artificial retina. Kybernetes: The International Journal of Systems and Cybernetics, 2002; 29(1): 31-52.
22. Cipolla, R., Hollinghurst, N., Gee, A. & Dowland, R. Computer vision in interactive robotics. Assembly Automation, 1996, 16(1): 18-24.
23. Roy, D. & Pentland, A. Learning words from sights and sounds: A computational model. Cognitive Science, 2002; 26(1), 113-146.