Towards the Design of Smaller Artificially Living Robots

Mark A. Lanthier and Franz Oppacher
Intelligent Systems Laboratory, School of Computer Science
Carleton University, Ottawa, Canada
<http://www.scs.carleton.ca/~lanthier>

Abstract

Recent research in the area of autonomous robots has concentrated on the development of simpler, smaller and cheaper robots. A robot is presented here which has been developed for the purpose of investigating the usefulness of robots with a minimal set of sensors. A carefully selected repertoire of combined instinctive behaviors allows the robot to remain functioning, providing the essence of an artificial life-form. The robot achieves landmark-based navigation for the purpose of finding energy to remain "alive". The programming of these behaviors is accomplished with hardwired networks of neurons that can be easily converted into electronic circuitry.

1 INTRODUCTION

Many autonomous robots are built with the intention of getting them to perform complex tasks. These complex tasks usually require complex sensors and a large amount of computation. A robot equipped with these sensors and computational processors is usually large and heavy due to the need for large power supplies. There is a growing amount of interest in the design of smaller and simpler robots employing nanotechnology [4], [3], [1], [11].

One possible method of reducing a robot's size is to replace the cluster of complex sensors and their processors with a minimal set of very simple sensors. This strategy cuts down on the amount of sensory information that needs to be processed and hence provides a reduction of the processing power requirements. A drawback of this approach is that the robot is limited to sensing simple environmental stimuli and thus it may have limited usefulness. It is our opinion that these robots can be most useful when many are used together in a cooperative manner.

There has been research in the application of neural networks [2], genetic algorithms [5] and classifier systems [8] to autonomous robots. The research often involved the learning of simple behaviors from scratch. That is, the robot started with no initial knowledge of its sensors and then through environment interaction, it was able to learn how to use its sensors to perform some simple behavior such as avoiding an obstacle, pushing a box or following an edge. With these techniques, the robot had to learn how to perform the behaviors over time. A problem with learning such simple behaviors is that the robot initially performs poorly during the learning process, and adequately only after a certain amount of time has elapsed.

Another approach to designing behaviors is to hardwire them as instincts so that the robot is immediately able to perform simple behaviors when initially placed in an environment. A benefit of instincts is that no extra processing power for learning is needed. Furthermore, the behavior can be easily coded in simple electronic circuitry which leads to a reduction in the amount of physical space required to implement a behavior mechanism. Furthermore, if many of these hardwired instinctive behaviors are to be incorporated into the robot, then they may all be coded onto one small electronic device. This reduction in processing power is of a tremendous advantage to nano-sized robots.

In this paper we describe an insect-like robot, RABI (Robotic Adaptive Behavioral Insect), which was developed in order to gain insight as to the usefulness of such simplified robots. A physical robot was built to evaluate the simplified design and a simulated version of the robot was used to study the various behaviors and

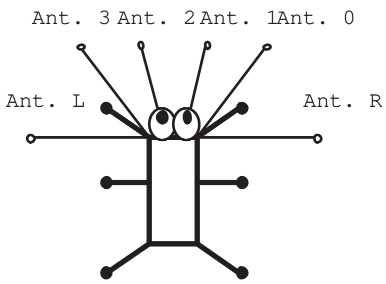

Figure 1: The antennae sensors of RABI.

motivational aspects corresponding to those of artificial life forms.

The sensors merely consist of 6 antennae which act as proximity sensors as shown in Figure 1. These antennae do not provide accurate proximity readings, but only a yes or no response indicating the presence or absence of an obstacle. The simulated version has additional simple sensors to detect light, energy, dirt, and a metal disk.

The rest of this paper describes various design aspects of building RABI. We first describe the implementation of the robot's behaviors and its motivation system that provide it with life-like characteristics. Simulation results show how the robot's behaviors can be combined to provide a robust overall system. We then briefly discuss the map building and navigational strategy used by the robot to locate and obtain energy.

2 IMPLEMENTATION OF BEHAVIORS

An approach similar to that of Beer [1] is used to program hardwired instincts into RABI. The instinctive behaviors are individually coded as networks of several interconnected neurons. The neurons are interconnected with excitatory (lines with arrows) and inhibitory (lines with circles) links to provide a method of low level computation. Our design of behaviors differs from Beer (1990) in that we use many different types of neurons in order to provide more functionality. The different types of neurons used are shown in Figure 2.

All neurons are similar in that they sum incoming signals and produce an output. The output of each neuron differs with its activation function. Some neurons, such as the *accumulative*, *sustain* and *differential*, keep an internal state according to previous input signals. In a sense, the network of neurons is a cross between a neural network and a state machine. All instinctive behaviors and control mechanisms (including leg coordination) are programmed solely with these networks.

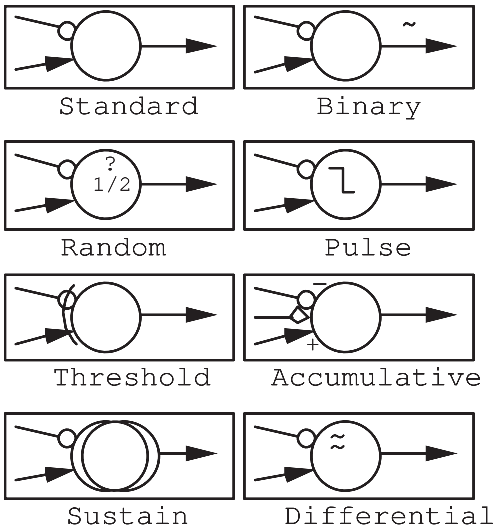

Figure 2: The different neurons used in the implementation of instinctive behaviors.

Some additional programming is currently being used for map-building and landmark matching. More information on the neurons and networks used is given in [6].

These neuron networks are used to coordinate leg movements and to implement various instincts of the robot. Instinctive behaviors are created for collision avoidance, wandering, edge following, vacancy (i.e. remaining in unoccupied areas of the environment), light seeking, energy seeking, cleaning, map building and landmark-based navigation.

Due to space constraints, only the leg coordination network and collision avoidance behavior are discussed here; for further details on the construction of the other networks, see [6]. The leg coordination networks of Beer [1] did not explicitly handle turning, and the timing for lifting and placing the foot down is not at all clear. A modified version of this network (with the new neurons discussed previously) is used to allow RABI to coordinate its 6 legs and walk using a tripod gait. The notion of timed pulses is eliminated since only one gait is desired. The network is shown in Figure 3 below and is duplicated for each of the 6 legs.

In this network, the FRT, BCK, FTUP and FTDN neurons are connected to the robots leg sensors and provide a binary response indicating whether the robot's leg is all the way forward, backward, up or down, respectively. The FTDN neuron instructs the robot to begin a stance phase (push forward) when the foot of the robot touches the ground by partially exciting the ST neuron. One additional neuron (not shown) is connected to the

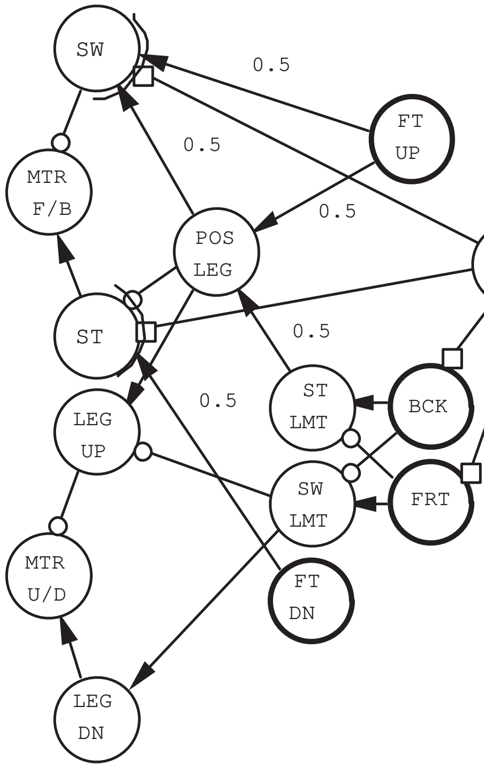

Figure 3: The neuron network for leg coordination.

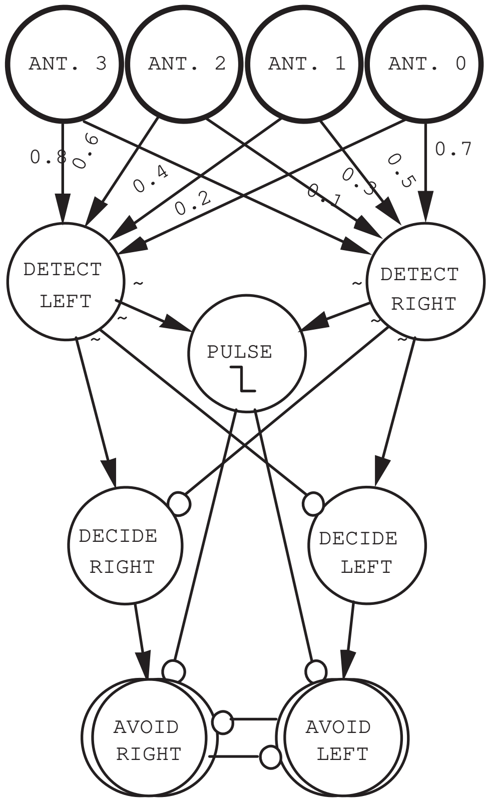

Figure 4: The neuron network for collision avoidance.

ST neuron of each leg to provide a constant partial excitation. Since the ST neuron is a threshold neuron, it is only fully excited when it receives 2 inputs of 0.5. Once the foot is lifted up, and the leg is all the way back, the FTUP and STLMT neurons excite the SW and LEGUP neuron and causes the robot to lift its leg and swing it forward. When the leg reaches all the way forward, the SWLMT neuron enables the LEGDN neuron and the robot puts its leg down. The MTRF/B and MTRU/D neurons are directly connected to the leg motors and cause the motor to spin in a CW or CCW direction depending on a positive or negative input. The role of the POSLEG neuron is to ensure that the leg stays in the swing mode until the leg is placed down or in the stance mode until the leg is all the way back.

Turning is accomplished by reversing the legs on one side of the robots body. Essentially, the robot pivots on the spot. The REV neuron provides a special excitatory input to the SW, ST, BCK and FRT neurons which causes them to negate their outputs. In the case of the SW and ST neurons, this reverses the direction of the motor. The BCK and FRT neurons provide a crossover of sensor data to compensate for reverse direction when excited. One additional piece of coordination is required to achieve walking. That is, the POSLEG neurons of adjacent legs inhibit each other. This is done similarly to Beer [1].

Figure 4 depicts the collision avoidance network. The topmost neurons represent the input from the frontal antennae (ANT0 being the rightmost). These neurons output binary data indicating whether or not the corresponding antenna is touching an obstacle. The DETECT LEFT and DETECT RIGHT neurons receive weighted input from each of the antennae as shown. From this weighted input, the DECIDE LEFT and DECIDE RIGHT neurons make a decision as to whether or not the robot should turn left or right. If the DETECT RIGHT neuron's input dominates the DETECT LEFT, then the robot decides to turn left, since there are more obstacles touching on the right side of the robot. The decision to turn left is made in a similar manner. One of the AVOID LEFT and AVOID RIGHT neurons is then enabled and it remains enabled until the PULSE neuron disables it. This occurs when the robot no longer detects a collision. As will be seen in the next section, the AVOID LEFT and AVOID RIGHT neurons compete with all other behaviors to decide which direction the robot should turn.

2.1 Motivation and Behavior Selection

In order for a robot to display some of the autonomous characteristics of living systems, it must be endowed with a repertoire of behaviors that give it a degree of unpredictability and randomness. The overall behavior of the robot should emerge as a function of the individual behaviors. This results in a more robust overall system [3],[1].

A robot with the ability to perform multiple behaviors may be faced with a decision as to which behaviors should be performed at any moment in time. In general, there may be conflicting behaviors that require the robot to perform two incompatible actions such as turning left and turning right. In this case, the robot must make a compromise by choosing which direction to turn. A simple method of avoiding conflicting behaviors is to allow the robot to give-in to only one behavior at a time as seen in [7]. If this approach is followed, then the robot should choose to exhibit the most important behavior which is defined in terms of motivational intensity. RABI is designed to adhere to 3 basic motivational rules, in order of importance:

1. Remain functioning (keep ample supply of energy)
2. Explore the environment for new sources of energy
3. Perform any tasks desired by the creator.

The rules are chosen so that the robot mimics an artificial life form. It exhibits a strong motivation to stay alive. RABI constantly monitors its energy level and when this becomes too low, it immediately attempts to find an energy source to ensure its survival. Initially, a robot placed in an unknown environment is unaware of the energy sources it contains. In order to find energy sources, the robot must explore, but it should not spend all of its time looking for energy sources since any useful robot would be required to perform some task for which it was designed. RABI instinctively does a lot of exploration when first placed into a new environment. After the environment has been mapped out and the robot has knowledge of some energy sources, this desire for exploration decreases. Eventually, RABI will cease to explore the environment, concentrating on the task at hand, occasionally taking a break to obtain energy.

In order to be able to make a decision as to which behavior is to be selected at any one time, RABI has hardwired priorities among the individual behaviors. This allows the most important behaviors such as energy seeking and obstacle avoidance to override the less important ones such as wandering and cleaning. The notion is similar to the subsumption architecture introduced by [3]. Figure 5 shows the connections required to implement this prioritized selection process.

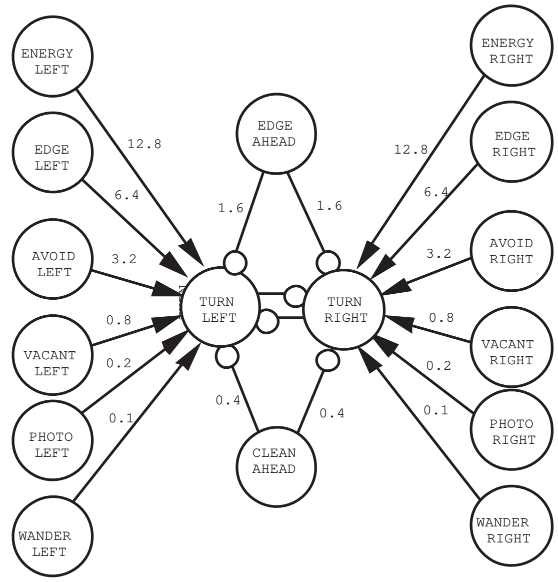

Figure 5: Connections of behavioral control neurons required for prioritized behavior selection.

Each individual behavior has 2 neurons whose outputs indicate that the behavior requires a left or right turn. The behaviors compete for overall control of the robot by connecting to the neurons responsible for actuator control (i.e. TURN LEFT and TURN RIGHT). The priorities are established by assigning different weights to the links from each behavior. Note that the weight values are in multiples of 2. This allows a high priority behavior to override all lower priority behaviors since the weight of an incoming high priority signal exceeds the sum of all lower priority incoming signals combined. With this strategy, many behaviors compete for control at any one time but only the behavior with highest priority is allowed to control the robot. The CLEAN AHEAD and EDGE AHEAD neurons correspond to moving forward and hence they inhibit the turning process.

With these connections it is possible to equip a robot with multiple behaviors. Furthermore, this strategy ensures that only one behavior will be active at a time. The behaviors are individually predictable, but by combining them, more complex behaviors emerge. Figure 6 shows the results of combining the wandering and vacancy behaviors. The dots represent the path that RABI has made within a simple square environment. With only a wandering behavior and the collision avoidance behavior (which is always enabled), the robot spends much of its time traveling along the environment boundaries. When the vacancy behavior is

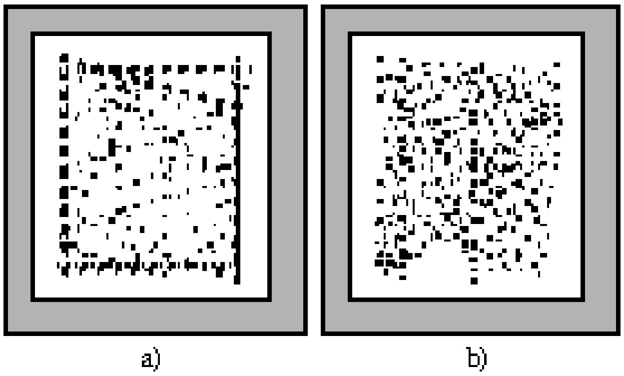

Figure 6: Robot's path when using a) just wandering behavior, b) both wandering and vacancy behaviors.

added, the path becomes more random since the robot spends most of its time away from the boundaries.

The use of multiple behaviors provides a degree of unpredictability and a more convincing representation of a living organism. The addition of a third behavior yields a more interesting result. Figure 7 shows the results of combining the wandering (W) and vacancy (V) behaviors with the light seeking (LS) behavior. A light source is now placed outside (beneath) the environment.

With solely a light seeking behavior, the robot's trajectory is very regular and uniform with circular patterns. The circular pattern arises from the implementation of the light seeking behavior: once the robot has passed the light source, it turns back around for another pass. By adding the wandering behavior, the robot is able to occasionally stray from the uniform path, but the overall regular movement pattern remains. With the addition of the vacancy behavior, the robot moves away from the wall so as not to rub up against it. This pattern is more random but the circular pattern still remains. With the addition of both wandering and vacancy behaviors, the resulting trajectory is quite random with almost no trace of the circular pattern. From these results, it is clear that by combining multiple behaviors, a more random behavior emerges. The resulting behavior is similar to that of a fly bouncing up against a window.

2.2 Cooperative Behaviors

It is also possible for these behaviors to complement one another through cooperation. Consider a robot that is equipped with a miniature scoop that drags on the ground beneath it. As the robot moves around, dirt morsels collect in the scoop and a simple sensor instructs the robot that the scoop is full of dirt and must

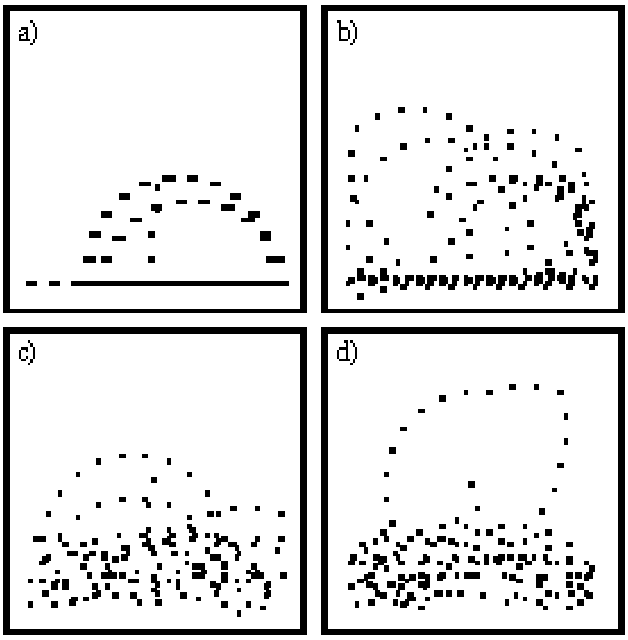

Figure 7: Randomness of trajectory for a robot with (a) only LS behavior, (b) LS and W behaviors, (c) LS and V behaviors and d) LS, W and V behaviors.

be emptied. The robot cold then proceed to the nearest wall and dump the dirt. This scenario represents the task of cleaning an environment. The simulated RABI has this ability through the use of its cleaning behavior. By wandering around in the environment, RABI is able to collect morsels of dirt and dump them along obstacle boundaries. Since the wandering is aimless, there are likely to be portions that remain uncleaned.

One simple method of improving the efficiency of this cleaning task is to somehow instruct the robot where to clean. If the robot is equipped with a light-seeking behavior, it could be attracted to a dirty part of the environment by placing a light source there. While it is near the light source, its dirt scoop becomes full and the cleaning behavior takes over. Once the scoop has been dumped, the light seeking behavior kicks in and the robot heads back to the light source. As a result, the area around the light source will be cleaned up. This is easily accomplished with RABI by enabling its light seeking and cleaning behaviors simultaneously.

One could imagine this technique employed in a colony of nanobots. The light source can be placed at different locations in the environment for short periods. The colony of robots would converge to this lit location and perform their cleaning task. Perhaps a second colony of robots could trace the environmental borders, picking up the dumped morsels, providing additional cleanup.

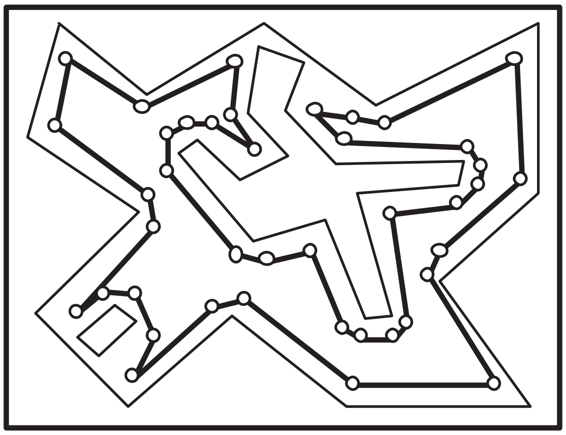

Figure 8: A map of a simple environment.

3 MAP BUILDING

If a robot is to remain functioning, it must have some method of finding energy sources. It should know when an energy source is nearby so that it can go to it and obtain energy. Assuming that the robot is equipped with some form of energy detection sensor (the sensor may detect fields of energy or wall sockets), then all that needs to be done to ensure survival is to remember the location of the energy sources. To do this, the robot must build a map of the environment so that the relative distance and direction to any energy source is always known.

RABI is able to create a map of an unknown environment using its 6 antennae and a marker (metal disk). It does this by tracing out obstacle boundaries, extracting distance and angle information from the obstacle's features. This method of mapping is known as landmark-based or feature-based mapping and has been previously studied by [10].

4 Landmark-Based Mapping

By simply tracing out obstacles, a robot can map out an environment in terms of edges and corners. Figure 8 shows an environmental map constructed with the simulated version of RABI.

In the map, the dark lines represent map edges and the circles represent corners. In order to build this map, the robot follows along the edges of the environment, recording the distances traveled between turns and the angles turned at each corner. When beginning the tracing process, the robot drops its marker. Once the robot arrives back at the location where the marker was dropped, it picks it back up and the trace is complete. By varying the minimum corner angle and edge length that can be detected, maps with differing levels of accuracy are produced. While the robot is tracing an obstacle, it records any energy source readings from its sensors and remembers which edges gave a strong energy reading. This allows the robot to associate edges and corners with energy sources. With this strategy, the robot could map out all obstacles in an environment.

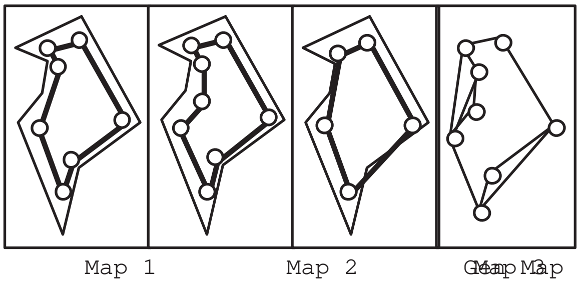

Figure 9: Generalized map created by matching and combining 3 separate mappings.

When an object is initially encountered, there is no immediate way of telling whether it has previously been mapped out. RABI keeps both a short and long term memory (STM and LTM respectively) containing simple neurons that represent edges and corners of an obstacle. The LTM keeps maps of all obstacle mappings made so far, and the STM keeps a sequence of 7 or 8 obstacle features. Whenever RABI encounters an obstacle, it begins tracing it and recording the features in the STM. Once this memory becomes full, it attempts to match its contents with a sequence in the LTM. If no match is found, the robot drops its marker and continues tracing the obstacle. If a match is found, then the location within the map is known and may be used for navigation purposes.

Due to the inherent inaccuracies of the robot's movements and measurements, the resulting map of an obstacle can differ upon each traversal. This presents a problem since the robot must have some way of matching 2 nearly similar mappings. When RABI obtains a new mapping of an obstacle, it compares the map with others in memory to see if they match within a small tolerance of edge lengths and turning angles. If so, it stores a generalized map representing a combination of the 2 maps. Figure 9 shows the generalized map resulting from a generalization of three separate obstacle maps.

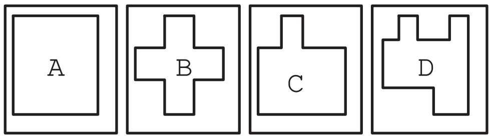

Figure 10: Simple environment examples.

4.1 Navigation on an Obstacle Border

In order to find energy sources, the robot must be able to navigate from edge to edge in the environment to reach the location at which an energy source was previously detected. Since the LTM is essentially a sequence of interconnected neurons, then once a partial sequence has been identified from STM, navigation merely requires the robot to travel to consecutive edges in the environment until the desired edge or corner has been reached. By spreading activation outwards from the desired location, the robot is given an indication as to which direction to head in (i.e. clockwise or counter clockwise along the obstacle border). The technique of spreading activation has been previously implemented by researchers such as [9].

The technique of matching STM sequences with LTM sequences is not at all without problems. In order for this method to work adequately, the environment must be sufficiently complex so that the obstacles provide enough information to distinguish between features. Figure 10 shows a set of 4 simple environments. With environments A and B, features would be hard to distinguish because of the similar edge lengths and corner angles within it. Thus the robot would not be able to determine its location. Environments C and D however, contain some distinguishable features which allow easy identification.

The method of feature-based mapping and landmark-based navigation is therefore limited to environments in which there is a large amount of variance among obstacle features and the obstacles themselves are sufficiently complex.

4.2 Navigation Between Obstacles

In general, there may be many obstacles in an environment. A robot may need to travel from obstacle to obstacle in order to reach a desired location. A robot equipped with only simple antennae sensors, should remain near obstacles since this is where its antennae are most useful. When traveling out in the open, the antennae do not provide any information. Thus, when navigating, the robot should only venture into open areas when absolutely necessary. Moreover, when in an open area, the robot should travel in a well defined path, such as a straight line, so that it is aware of where it will be once another obstacle is encountered.

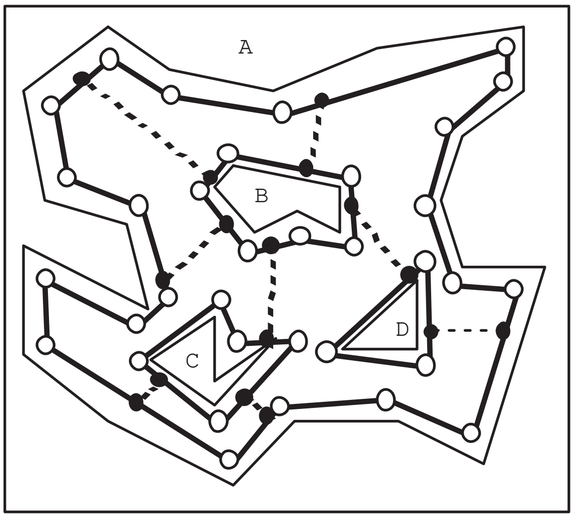

Figure 11: Additional links required to store adjacency information between obstacles.

When building a map of the environment, RABI also records the adjacency information between 2 obstacles by adding an imaginary edge between them. Many of these imaginary edges are created so that the robot has an indication as to the relative distances and directions between obstacles. Figure 11 shows a simple environment with the imaginary links added. These links are added by simply adding additional neurons connecting edges in the LTM. With these links, RABI knows that if it turns away at some point along an obstacle boundary at some angle, then by traveling straight, it will reach a known edge of another obstacle. By using the spreading activation technique with this revised LTM, it knows which direction to travel along the obstacle as well as when to turn away and head off into open space to get to another obstacle. As a result, RABI is capable of point to point navigation from obstacle to obstacle which is used for the purpose of finding previously encountered energy sources.

5 CONCLUSION

It is clear that a reduction of sensor size and complexity leads to overall reduction of the autonomous system and reduces the processing requirements. We have shown that by using the notion of motivation-controlled behaviors a robot can be built to remain functioning as well as performing some task. The robot presented is given a repertoire of simple behaviors which are programmed using a hardwired network of neurons. The method used to construct such behaviors allows a further reduction of the robot's size and weight. Additional programming was also used to provide the robot with point to point landmark-based navigation, allowing it to locate and ingest energy sources. The research reported here takes a first step towards the development of small, light-weight and inexpensive artificially living robots.

References

[1] Randall D. Beer, "Intelligence as Adaptive Behavior", Academic Press, London, 1990.

[2] Randall D. Beer, and John C. Gallagher, *Evolving Dynamical Neural Networks*, Adaptive Behavior , MIT Press, London, 1992, pp. 91-122.

[3] Rodney A. Brooks, *A Layered Intelligent Control System for a Mobile Robot*, 3rd International Symposium on Robotics Research, Vol. 3, MIT Press, London, 1986, pp. 365-372.

[4] Anita M. Flynn, *Gnat Robots*, AI Expert, Vol.2, No.12, Dec. 1987, pp. 34-41.

[5] John R. Koza and James P. Rice, *Automatic Programming of Robots using Genetic Programming*, Proc. 10th National Conf. on A.I., San Jose, MIT Press, London, July 1992, pp. 194-201.

[6] Mark A. Lanthier, *Implementation of Adaptive Behaviors in a Simple Insect-like Robot*, Masters Thesis, Carleton University, Ottawa, Canada, 1993.

[7] Pattie Maes, *A Bottom-Up Mechanism for Behavior Selection in an Artificial Creature*, From Animals to Animats: Proc. of the 1st Int. Conf. on Simulation of Adaptive Behavior, MIT Press, London, 1991, pp238-246.

[8] Sridhar Mahadevan and Jonathan Connell, *Automatic Programming of Behavior-Based Robots using Reinforcement Learning*, Artificial Intelligence, Vol.55, Elsevier, 1992, pp. 311-365.

[9] Maja J. Mataric, *Navigating With a Rat Brain: A Neurobiologically-Inspired Model for Robot Spatial Representation*, From Animals to Animats: Proc. of the 1st Int. Conf. on Simulation of Adaptive Behavior, MIT Press, London, 1991, pp. 169-175.

[10] Ulrich Nehmzow and Tim Smithers, *Mapbuilding Using Self-Organizing Networks in "Really Useful Robots"*. From Animals to Animats: Proc. of the 1st Int. Conf. on Simulation of Adaptive Behavior, MIT Press, London, 1991, pp. 152-159.

[11] Mark Tilden, *Photovores*, Scientific American, September 1992, pp.42.