EVOLVING SENSOR MORPHOLOGY ON A LEGGED ROBOT IN NICHE ENVIRONMENTS

GARY B. PARKER, CONNECTICUT COLLEGE, USA. PARKER@CONNCOLL.EDU
PRAMOD J. NATHAN, CONNECTICUT COLLEGE, USA. PJNAT@CONNCOLL.EDU

ABSTRACT

This paper discusses the issue of evolution of morphology and automatic design, specifically evolving sensor morphology on a legged hexapod robot in niche environments. The evolution of sensor morphology in different environments, in particular, type of sensor, angle of heading and its effect on controller complexity for a simulated hexapod robot are described. This automatic design method enables the system to decipher relevant stimuli in an environment, increases the efficiency of the robot and also indirectly alters the controller of the robot to take advantage of the characteristics of a given environment.

KEYWORDS: Sensor Evolution, Automatic Design, Artificial Life, Genetic Algorithm, Sensor Optimization, Evolving Morphology

1. INTRODUCTION

The evolution of robot morphology is a practical method in the search for a successful agent for a given environment. Sensors play a key role in deciphering the environment an agent has to perform in, since an agent is limited by the range of stimuli it can detect, as well as the effectiveness of the sensors that at are at it's disposal. Due to this, the limitations of sensors are co-related with the agent's capability to learn in the environment. The effectiveness of these sensors in interpreting the environment is dependent on features such as positioning, the types of sensors, the manner in which the sensors are integrated into the controller of an agent, the accuracy of the sensors and finally the noise tolerance. Each of these factors has to be integrated in the agent in such a way as to allow the agent to deal with the environment at hand while avoiding redundancy in the information it obtains from the environment.

Recent studies in evolving sensor morphology deal with creation of adaptive hardware to evolve the positioning [1] or and array of the same type of sensors [2, 3]. There exists a limitation to this approach since using a single type of sensor does not provide the control system a wide variety of information about the environment thus reducing the capabilities of the agent as a whole. Other research pertained to assembling structures from modular parts [4]. Research done on Tinkerbots [5] used the idea of re-usable sub-procedures to design scalable complex designs. Yet as pointed out by Pollack et al, [6] these agents do not interact with the environment due to the lack of sensors. We believe this lack of sensors do not allow these agents to evolve beyond a certain complexity due to the lack of constant feedback from the environment. Research conducted by Bugajska and Schultz [7] and Mautner and Belew [8] shows strategies to co-evolve the robot controller and the sensor morphology. The former study dealt with finding general sensor morphology for any environment configuration, whereas in the latter the complexity of the environment was unchanged. Moreover, Bugajska and Schultz did not utilize different types of sensors or utilize stimuli other than obstacles and distance to goal. In addition, unlike their research [7], this paper deals with evolving sensors with pre-defined capabilities of range and coverage. This is an important difference since the sensors available in the market are not reconfigurable in terms of changing their ranges and areas of coverage. We argue that, even with this restriction, the positioning of the sensor on an agent plays a key role in deciding how many sensors are needed for a given task. The location of a sensor also enables the agent to take full

advantage of the terrain to complete a given task. Moreover, this system is more practical and is easily transferable to a real robot, as will be done in future work. Since this study involves evolving sensor morphology, one of the domains the study covers is deciphering the information obtained from the environment to complete a task.

Pfeifer [9] and Atkins [10] suggested that there is a relationship between the morphology, the controller and the environment of a given system. For a given task some stimuli provided by the environment are useful while other stimuli are not. The task of a sensor is to sense these stimuli. The more complex an environment is the more complex the sensors need to be to obtain and extract relevant stimuli in an environment. At the same time, the agent has to decipher which stimuli are necessary for the completion of a certain task and prune the unnecessary ones. Pruning makes the agent more efficient, as extra sensors put a strain on the energy supply of the agent, and allows for a less complex controller that does not use all the stimuli available. Evolving sensor morphology allows for optimization in terms of power consumption, controller complexity and reaction time. In this paper we show that evolving sensor morphology allows for the automatic design of an agent that can exploit the environment to complete a given task using stimuli in the environment as cues, decipher relevant information to complete a task, discard irrelevant information not needed to complete a given task, and decrease the number of sensors it uses for a given task, in a niche environment. To investigate this method we use a simulation of the real world system described in Section 2.

2. NAVIGATION TASK

This paper is part of a larger study that uses a genetic algorithm (GA) in the search for a successful sensor morphology and controller for a given robot. The aim of this part of the study is to evolve a sensor morphology which exploits the characteristics of an environment to complete its task given a fixed control program. This problem focuses on minimizing the sensors used and changing the sensors' angles of heading, to exploit a given environment and task. The problem does not pertain to finding a generalized morphology (as in [7]) but rather focuses on using GAs to find specialized agents for niche environments [9].

The Task: To traverse from a start position (20, 100) to the target (270, 160).

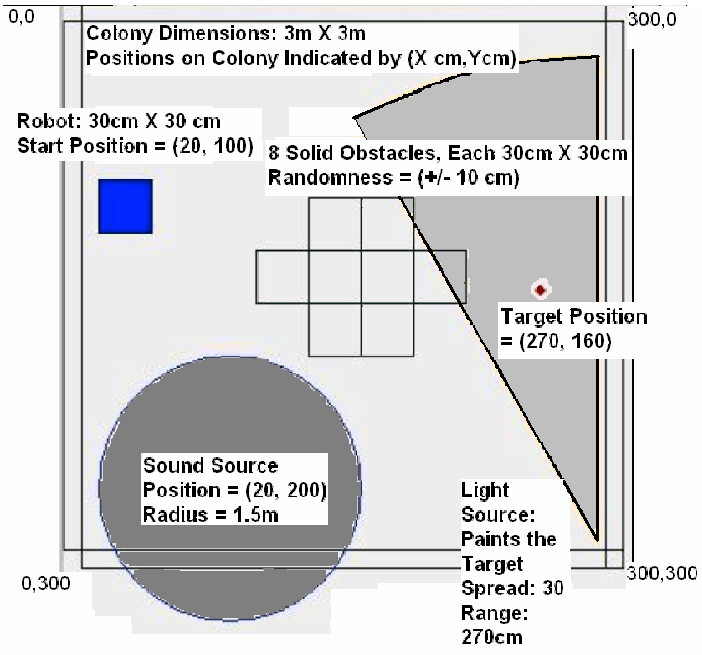


Figure 1: Environment in Central Mountain configuration.

The Niche Environments (Figure 1: Central Mountain configuration): A 3m x 3m walled area established in the Connecticut College Robotics Laboratory with 30cm x 30cm x 5cm solid obstacles, a light source and a sound emitter. The solid obstacles are not high enough to block the light beam or muffle the sound. The obstacles and the walls can be detected by the tactile sensors placed on the robot. The solid obstacles are configured in 7 different configurations (Figure 5). There is a random variable involved in the placement of the solid obstacles to allow the results to have a level of generality. For each generation the obstacles were moved randomly by a factor of +/- 10cm in the X and the Y direction. The agent is also evolved in a completely random environment, in which the solid obstacles are moved at random for every test run. The obstacle density was 8%. The target is a fixed point within the area illuminated by a light.

The Agent: The ServoBot [11] is a hexapod that has 2 degrees of freedom per leg. The robot is equipped with a sensor base, which is attached to the top of the robot and completely covers the robot so that the actuators and effectors of the robot do not affect the sensors' field of view. This base is 30cm x 30cm and acts as an easily reconfigurable platform for the sensors. The agent's locomotion is non-deterministic, i.e. given a start position and orientation, the end position after a step has a degree of randomness involved. This non-determinism in the gait is due to the inconsistencies of the robot construction and inherent noise caused by having a legged robot interact with the world. The agent itself has no information about the location of the target point. To navigate it must learn and use cues from the environment to its advantage.

The Sensors: The sensor base carries on it modules of 4 light sensors, 4 sound sensors and 4 tactile sensors (12 sensors). The sensors are all binary; they either detect a stimulus or do not. Each of the sensors is a reusable module that can interface with the controller. The position and the angles at which sensors point can be changed on the real robot, but for this paper we only evolve the angles at which the sensor point (heading angle). This approach allows the sensory information to be complex enough for the robot to be successful in the environment while making the simplifications necessary to allow evolution [6]. The placement of the sensors on the sensor base is shown in Figure 2. The sensors are placed facing outward, with the tactile sensors placed in the corners of the Sensor base and the light and sound sensors placed on the edges of the sensor base. Range and spread of the sensors are given by the triangles shown in the Figure 2. A sensor registers true if any part of its range/span is in the area of its particular stimulus. (Light for the light sensor; Sound for the sound sensor; Walls and solid obstacles for the tactile sensors). Specifications indicated that stimulus coming from straight ahead showed unreliable data, giving the sound sensor its wide triangular form.

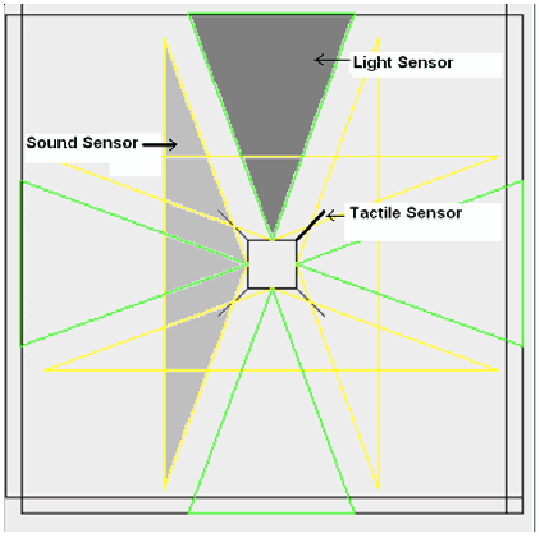


Figure 2: Sensor Placement on Sensor Base and Ranges of Sensors

The Controller: The agent is controlled by two Basic Stamp II modules, a primary module to control the locomotion of the ServoBot, a secondary module to get information from the sensor and feed this information into the first Basic Stamp II module. The space constraints on the Basic Stamp II restrict the controller complexity and limit the number of sensors used to 12. The controller consists of 14 simple if ... then rules that trigger one of 16 turn gaits of the ServoBot, which are of the form: *If (Sensor detects a stimulus) then (Trigger gait number X)* The 16 gaits are previously developed subroutines that change the direction and heading of the robot and accurately represent the movement of a specific ServoBot... The default gait enables the ServoBot to walk straight, but due to inconsistencies in the construction of the robot, it turns slightly towards the left. There is only one reverse gait, all the other gaits move the ServoBot forward and change the heading and lateral position of the robot by varying degrees [11]. The controller is predefined and not evolved by the GA, but the selection of the sensors by the GA affects the controller as described previously. Rule ordering conflicts are circumvented by predefining a rule order. The rules priorities are as follows: tactile sensors are given first priority, sound sensors are given second priority and light sensors are given third priority.

3. EVOLUTION OF SENSOR MORPHOLOGY AND TESTS

The parameters that are evolved in this paper are the heading angle of the sensors and which sensors to keep running during the length of the run; the position of the sensors on the sensor based are fixed as shown in Figure 2 and do not change. The heading of the sensors can be rotated 360 and as many as all 12 sensors can be turned off. The chromosome used was 120 bits long, with the first 12 bits representing which sensors to put on or off and the remaining 108 bits representing the angles at which of the each of the 12 sensors are placed on board the robot base. The chromosome was divided into 1 set of 12 bits and 12 sets of 9 bits. During evolution, each set underwent a single point crossover using stochastic (roulette wheel) selection of the parents and had a mutation rate of 4%. We ran 128 individuals for 128 generations for each environment. This was repeated 5 times with random starting populations to check for consistency of the results. The individual with the highest fitness was added to the next generation without change.

The agent was given 100 time intervals to complete the task. A time interval is the time it takes a robot to complete one step. A step starts with the legs in a ready to step position (right front and back legs and left middle leg forward; remaining legs back) and returns to this position after a full step cycle. This time is the same for all 16 step cycles. The time intervals continue to elapse even if the robot collides with an obstacle and tries to continue to move but it takes up time steps. A collision does not stop a test run. The agent can get out of a collision since the agent can shake free of the obstacle and continue its task due to its non-deterministic gait. The run is stopped if the target position is reached or when the agent runs out of time (100 time intervals).

```
If (Agent reached Target)
   Fitness = 100(2(Number_Sensors_Off) + (Total_Time - Time_to_Target))
Else
   Fitness = 100(Number_Sensors_Off + Distance_to_Target)
```

Figure 3: The Fitness Function

The fitness function (Figure 3) is dependant on the success of the agent. If the agent is successful in finding the target, the fitness is based on the number of sensors it has off and the amount of time it took the agent to get to the target. If the agent is unsuccessful in reaching the target, the fitness is dependent on how far away the robot is from the goal as well as the number of sensors it has off. In the first case, to achieve the maximum fitness the robot has to have all its sensors off and reach the goal without any time being used, which is an impossible scenario in the simulation. A successful agent has the theoretical maximum fitness of 12400 while an

unsuccessful agent has a maximum fitness of only 5440. Even though both parts use the number of sensors put off, it is given only half the weight when the agent is unsuccessful.

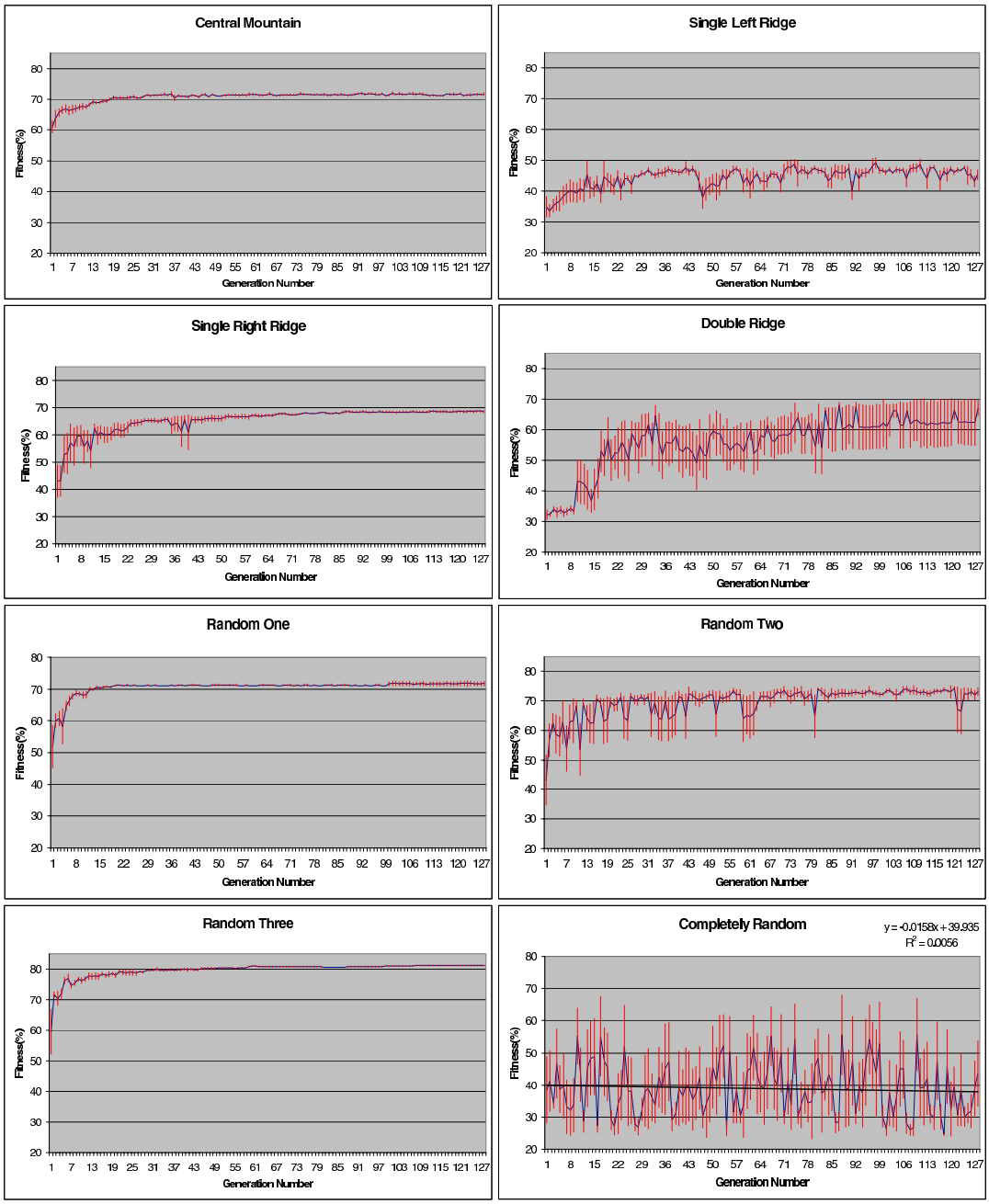


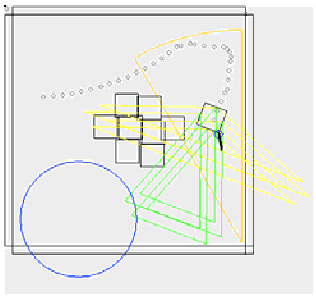
Figure 4: Fitness growth of elite individuals averaged over 5 test runs.

4. RESULTS AND DISCUSSION

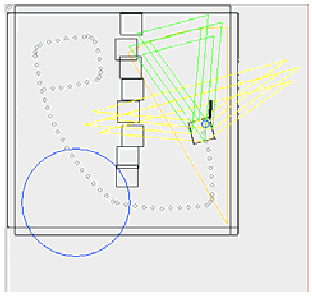
The results in Figure 4 show the average of the elites over 5 runs (Elite: individual with the topmost fitness for a given generation number). The error bars are the standard deviation, showing how consistent the results are over the 5 runs.

In all of the environments, the GA improved the fitness of the agents. In the case of the completely random environment, there is no overall increase in fitness over the number of generations (linear regression shows a small negative slope of 0.0158). This environment showed

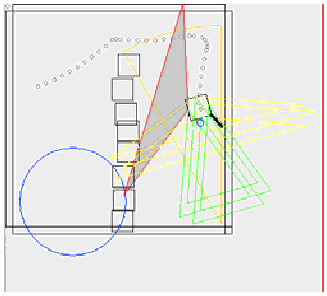
that if the environment is completely random, the success of the individual too is completely random. The average fitness in the case of the completely random environment was 38.91419 % of the total obtainable fitness. It is interesting to note that environments with a starting fitness of less than 38.9%, namely Single Left Ridge and Double Ridge, had results that were much less consistent in the second half of the generations than the ones that started with fitness above this value. The initial values of the standard deviation in all cases were high due to the random generation of the initial population.



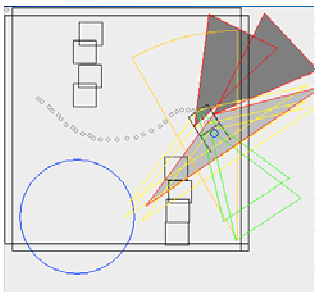
Central Mountain
1 Tactile Sensor at 330º



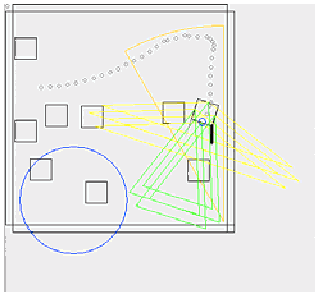
Single Left Ridge
1 Tactile Sensor at 18º



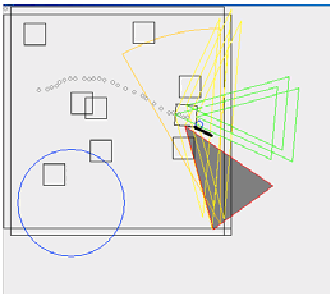
Single Right Ridge
1 Tactile Sensor at 333º
1 Sound Sensor at 123º



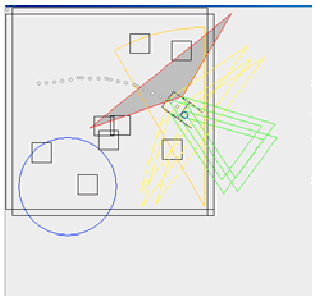
Double Ridge
1 Sound Sensor at 359º
2 Light Sensors at 240º and 260º



Random One
1 Tactile sensor at 342º



Random Two
1 Tactile Sensor at 23º
1 Light Sensor at 49º



Random Three
1 Sound sensor at 195º

Figure 5: Paths taken by elites. The sensors that were used are shown (darkened and filled in for ease of identification) along with the angle they were placed relative to the robot base. The unused sensors all point north relative to the robot base.

In Random Two the relatively high deviations towards the end of the run (generation 122, 123) are due to dips in the fitness. The dips can be attributed to the randomness of the environment as described in Section 2. Due to this randomness, an individual that has performed well in one generation might not do so in the next if the evolved solution is not general enough.

The randomness ensured that a solution which was not general enough would not succeed in the following generations. As the agent evolved, individuals that worked with the generality in the environment were generated. In most cases, as the as the population evolved over the number of generations the results became more consistent thus showing that the solutions were robust enough to work in this level of generality.

The paths taken by a sample of the elites during the final run are shown in Figure 5. The final agents that were obtained by this method could decipher relevant stimuli needed to complete a task without being redundant (hence more efficient), using a low number of sensors. It is important to note that the individual agent does not have information on its location in the environment or that of the target's location. It must find the target using characteristics of the environment that are helpful to find this target point, and try to minimize the number of steps used to get to this point.

The success of the agent was heavily dependant on the environment as some graphs show that the solution was more difficult than others. In some environments the path taken to reach the goal was far less than optimal especially in Single Left Ridge. This result was partly due to the agent's build which made the agent drift left during its straight walking gait. Yet most results showed fairly efficient paths taken to the target point. This difference in path efficiency can be attributed to the agent not having information of the target's location, except though cues in the environment, hence the different configurations of the environment worked to its advantage or disadvantage. This dependence on environment configuration is most apparent in the differences between Single Left Ridge and Single Right Ridge, where we see that due to the ServoBot's tendency to head left, it takes a round-about path in the former environment, but the agent does markedly better in the latter environment.

In the case of Double Ridge, Random Two, and Random Three the paths taken were very close to the shortest paths obtainable, given the gaits of the ServoBot.

In the case of Central Mountain, Single Left Ridge, Single Right Ridge, Random Two, and Random Three the light sensor was not used at all, even though the target was in the area of the light and light is the most apparent way to find the target. However, this is not necessarily the best way to find the target. Using the light stimulus is redundant if the ServoBot could find the target without it and light sensors might not be the best way to avoid obstacles. The Central Mountain run shows this most explicitly. Here the light sensors were not used because it would cause a collision with the Central Mountain. The strategy that was evolved instead made use of its tactile sensor to navigate to the point which minimized the sensor usage and consistently allowed the agent to complete the task.

In terms of complexity of the controller, at most three sensors were used by the final agents for the environments. This corresponds to only three if…then statements being used, as opposed to the original 14. This is a significant difference especially since the controller used on the actual ServoBot is a BASIC Stamp II with limited processing speed and memory. Moreover, the reduction of the sensors used significantly reduces the number of pins used on the controller and decreases the power consumption of the robot.

5. CONCLUSION AND FUTURE WORK

The results of learning with a GA shows that evolving the sensor morphology of an agent is a suitable strategy for designing an agent for niche environments. The evolution also made the robot and controller efficient and pruned the number of rules necessary for completing the task.

The agent completed its task for every environment consistently at the end of the runs without having sensors that explicitly directed it to the target point. Instead it used the stimuli provided by the environment to navigate to this point. This shows that adaptable sensor morphology is beneficial for designing an agent to complete a given task for a given environment. The results also show that, given the limitations of the study so far, this strategy is not suitable to design an agent capable of completing its task in a highly general environment. The evolution of sensor

morphology on the legged robot allows the robot to learn its environment and pick out cues needed for completing its task, hence when these cues are random (Random Environment) the success of the robot is also random. The results essentially show evolved Braitenberg-like vehicles. As is the nature of such reactive systems, the behavior of such an agent is fairly simple but very efficient for simple tasks such as used in this paper. The study shows that the configuration of the environment is a very important factor in deciding which sensors and stimuli to utilize. An important factor in automatic design using evolutionary computation is the ability to decipher and utilize relevant information in the environment. This ability is highly dependent on the environment and task of the evolving agent. The final product of the GA was an agent that was a specific solution to a specific environment. The high priority given for efficiency ensured that the solution would not be general and the final designs of the agents hinted that as an agent's efficiency increased, the generality of its navigation decreased.

Future work will include randomizing the robot's start position and heading to some degree in the hopes of getting a more generalized solution for a specific niche. In further research, the sensor morphology and the controller will be co-evolved. For this part of the study, both the morphology and the controller will use the idea of modularity by utilizing re-usable sub-routines and sensor modules. The obtained results will be tested on an actual robot operating in a real environment to see how accurately the results of the simulation transfer to the real world.

6. REFERENCES

[1] K. Balakrisha and V. Honavar, "On Sensor Evolution in Robotics," Proceedings of the First Annual Conference on Genetic Programming, Stanford University, USA, 1996.

[2] M. Fend, M. Yokoi, R. Pfeifer, "Optimal Morphology of a Biologically-Inspired Whisker Array on an Obstacle-Avoiding Robot," Proceedings of the 7th European Conference on Artificial Life, ECAL Dortmund, Germany, 2003.

[3] L. Lichtensteiger, "Towards Optimal Sensor Morphology for Specific Tasks: Evolution of an Artificial Compound Eye for Estimating Time to Contact," In Sensor Fusion and Decentralized Control in Robotic Systems III, Proceedings of SPIE Vol. 4196, pp. 138-146, 2000.

[4] P. Funes and J. Pollack, "Computer Evolution of Buildable Objects" Proceddings of the Fourth European Conference on Artificial Life, Cambridge, USA, 1997.

[5] G. Hornby, H. Lipson, and J. Pollack, "Evolution of Generative Design Systems for Modular Physical Robots," Proceedings of the IEEE International Conference on Robotics and Automation, pages 4146–4151, 2001.

[6] J. Pollack, H. Lipson, G. Hornby, and P. Funes, "Three Generations of Automatically Designed Robots," *Artificial Life* Vol. 7, Issue 3, pages 215 – 223, 2001.

[7] M. Bugajska, and A. Schultz, "Coevolution of Form and Function in the Design of Micro Air Vehicles," Proceedings of the NASA/DoD Conference on Evolvable Hardware, Alexandria, VA, USA, 2002.

[8] C. Mautner and R. Belew, "Evolving Robot Morphology and Control," Proceedings of Artificial Life and Robotics, Oita, 1999.

[9] R. Pfeifer, "Building 'Fungus Eaters': Design Principles of Autonomous Agents," Proceedings of the Forth International Conference on Simulation of Adaptive Behavior, Cambridge, USA, 1996.

[10] M. Atkins and P. Cohen, "Monitoring Strategies of Embedded Agents: Experiments and Analysis," Proceedings from Animals to Animats 4, Cambridge, USA, 1996.

[11] G. Parker, "Evolving Cyclic Control for a Hexapod Performing Area Coverage," Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, Banff, Canada, 2001.