Spatial Memory Structures for Sensor-Guided Robot Navigation *

Karthik Balakrishnan and Vasant Honavar †
Artificial Intelligence Research Group
Department of Computer Science
Iowa State University, Ames, IA - 50011, USA
balakris@cs.iastate.edu, honavar@cs.iastate.edu
(515)-294-1098

Abstract

Although evolutionary algorithms offer an attractive and versatile approach to the automated design of behavior and control structures for mobile robots, they cannot anticipate the detailed structure of specific environments that the robot might have to deal with. Robots must thus possess mechanisms to adapt to the environments they encounter. In particular, mobile robots need stuctures for building and using spatial maps to aid in the successful exploration and navigation of *a-priori* unknown environments. This paper proposes a biologically inspired computational model for the acquisition and use of spatial memory structures for mobile robot navigation. Preliminary experimental results indicate that the proposed mechanisms can be effectively exploited by evolution in the design of high-performance robots.

1 Introduction

In the last four decades, mobile robotics has progressed significantly through advances in sensor technology, sensory information processing, VLSI technology, and the development of new effectors. For instance, accurate laser rangefinders have been developed, and a variety of devices for sensing anything from light intensities, to acceleration, to the slightest degree of movement have been invented [13]. With the advances in low-cost vision technology, improved algorithms for sensory processing, information fusion from multiple sensors, better image processing and recognition algorithms, and robust robot path-planning algorithms have arisen, leading to mobile robot applications in obstacle-avoidance, wall-following, light-tracking, homing, and recent applications like mail-delivery, outdoor driving (autonomous land vehicles (ALVs)), and *sojourner* (the Mars rover).

In spite of these advances, practical mobile robotics has to surmount many problems in order to be effective, primarily the unpredictability and noise of the real-world and the imprecision and operational noise inherent in the various robotic components. In order to successfully interact with dynamic, and often novel environments, the robot needs to be equipped with a suitable set of learning and adaptation mechanisms. Task environments that robots typically deal with offer particular challenges to learning in terms of the form and amount of information that is available [11, 1]. Reinforcement learning, which only requires scalar feedback, has found use in learning *policy* or *action functions* (the right set of actions to perform in each sensory state) for robots [11, 8, 10]. Availability of an internal model of the environment often simplifies the learning of action functions [11, 18].

Robots that have to navigate and manipulate objects in space can benefit from structures for acquiring and using internal models or *spatial maps* of their environments. Such models can be of great value in the identification and avoidance of obstacles, picking up specific objects and depositing them at target locations, homing in on goal locations, etc. For instance, a robot might learn a spatial map of obstacles in its environment and use it to navigate to specific goal locations efficiently.

A number of researchers have argued for the use of such spatial maps in mobile robot navigation. Brooks [7] argued for the use of *topological maps*, which represent the world as an abstract network of nodes and arcs. The nodes

---

[*] Submitted to GP-97

[†] This research is partially supported by the National Science Foundation (through grants IRI-9409580 and IRI-9643299) and the John Deere Foundation. Scalable Computing Laboratory of the DOE Ames Laboratory at Iowa State University provided some of the computing resources used for the experiments reported here. We would like to thank the PGAPack group led by David Levine (formerly of Argonne National Labs) for making the PGAPack software available and Brian Walenz for making several modifications to PGAPack.

correspond to *distinctive* places while the arcs between nodes represent the paths between the distinctive places. Such maps do not contain any metric (or geometric) information; only notions of proximity and order, and are thus robust to local movement errors. In addition, topological maps represent space very compactly since they only represent distinctive places and not the whole world. Many variations of such topological maps exist [6, 17, 20, 16, 12, 30]. However, the significant issues of what corresponds to a distinctive place, and how the map is to be represented and used, still remain open research problems.

Animals, on the other hand, are capable of highly sophisticated spatial navigation behaviors, and in many cases, do so with a great deal of ease. For instance, ants, bees, rodents, doves, etc., demonstrate a suprising sense of space, and are capable of locating their homes even after pseudo-random movements in search of food or their young [21, 14]. In the cognitive and neuroscience communities, it is stongly believed that these animals construct and use some form of a *cognitive map* to aid their spatial navigation needs. The hippocampal formation has been known to play a crucial role in the spatial learning ability of animals and has a suspected involvement in the acquisition and representation of ,the cognitive map [25]. However, even after decades of research, the exact nature of the spatial information encoded and the representation and use of the cognitive map remains unanswered in its entirety.

In this paper, we develop a model of spatial learning and navigation, which is strongly inspired by current neurobiology literature yet is highly applicable to contemporary robotics. We evolve neurocontrollers for robots which have this spatial learning model built-in. The robot's task is to clear a square arena by pushing boxes to the enclosing walls. However, the robot has limited battery power which is consumed by the sensors, the neurocontroller units, and the robot effectors (like the drive train and the wheels). It thus needs to charge up by approaching power sources. During the execution of its task the robot might come across power sources, which are represented in its spatial map as distinctive places. It can then return to the nearest power source (if it stores metric information also) when it is low on battery power. Much of this research is currently in progress and we only present some preliminary results here.

2 Aspects of Spatial Navigation in Animals

For most animals, locomotion (and hence navigation) of some form is an essential part of its behavioral repertoire, which is required to find food, avoid predators, find mates, etc. Animals appear to perform this task seemingly effortlessly, and in many cases, with a great deal of enterprise. One of the primary behaviors demonstrated by animals, is *dead reckoning*. The term is thought to be a corruption of *deduced reckoning*, and refers to the process of updating one's estimate of one's position on the basis of knowledge of how fast one has been moving, in what direction, and for how long. There is considerable evidence that this process plays a fundamental role in navigation in animals from ants to humans [14]. For instance the desert ant, *Cataglyphis bicolor*, follows a tortuous path in search of food and wanders as far away as 100 meters from its nest. During this exploratory search, its turning velocity attains rates of 4000 degrees per second. In spite of these, upon finding food, the ant turns and heads directly for home at a brisk 15 meters per minute, successfully locating the 1 mm nest opening in the featureless desert [14]. Studies with rats and gerbils have also confirmed the dead reckoning ability of rodents [14].

In the neuroscience community, the hippocampal formation and adjacent cortical regions of the medial temporal lobe have long been associated with spatial learning and memory. Hippocampal lesions have been known to impair spatial learning skills in humans and animals. Further, evidence from single-unit recordings and lesion studies in nonprimate mammals, mostly rodents, have also pointed to the significant involvement of the hippocampal formation in spatial learning and reasoning tasks [19]. It is believed that in such cases the hippocampus represents spatial information in the form of cognitive maps [25] which are then used for efficient, purposeful, goal-directed navigation. However, in spite of these insights and the many models of hippocampal function that have been proposed, the precise nature of the hippocampal code and the nature and use of cognitive maps remains to be fully understood.

Advances in the neurosciences have uncovered several key pieces of information regarding biological information processing.

1. Pyramidal cells in the regions CA1 and CA3 of the rat hippocampus are mostly *place cells*, that is they fire only when the rat is in a particular portion of its environment [24, 25]. Thus, place cells represent an ensemble code for the region of the environment [29]. However, the exact nature of the code is still open to debate.

2. *Head-direction cells* have been found in several regions of the brain, including the dorsal presubiculum and the posterior neocortex [27, 28]. These cells appear to fire preferentially based on the current orientation of the head. This enables the animal to determine the *egocentric* (with respect to the animal) headings of visible landmarks.

3. Behavioral experiments suggest that the motor system plays a very important role in the formation of cognitive maps in rats [23]. For instance, when rats were wrapped in a towel, restraining their motor activity entirely, almost all hippocampal neuronal activity ceased. Neuronal activity did not resume even when the rats were passively moved through the environment to regions that resulted in place cell firing earlier. Also, if the animal is placed in a familiar environment and then the lights are switched off, place cell firing continues in the darkness, in the absence of visual cues. These suggest that the motor system input is somehow a critical determinant of place cell firing.

Some or all of these facts have been used to suggest hippocampus-aided navigation models for animals [9, 22, 26]. However, none of these models explain the neurophysiological and behavioral data in their entirety. For instance, the model of Burgess et al. [9] aims to explain the hippocampal place cell data (including their relationship to EEG θ-rhythms) and head-direction cells. However, it falls short of demonstrating dead reckoning related behavior, including the ability of animals to navigate in darkness in familiar environments. The model of McNaughton et al. [22] is strongly influenced by the dead reckoning behavior of animals and the presence of head-direction cells. In their model, place cells encode landmarks in the form of a vector representation. However, they do not have an implementation of this theory. A number of issues need to be handled in order to make this model implementable, especially the vector representation to be used and the associative network for computing the new headings. Further, in order to determine a path to the goal location, this model assumes the existence of a landmark that is visible from the current location as well as the goal. Redish and Touretzky [26] have a theory of rodent navigation that is consistent with a number of neurophysiological and behavioral data. Their place cell theory is very much like ours (ref. section 3.1). However, unlike our model, their model does not produce navigational signals.

3 A Neurobiologically-Inspired Spatial Navigation Model for Robots

We have developed a model of spatial navigation that incorporates neurophysiological as well as behavioral principles of animal spatial behavior. Further, our model can be conveniently used in robotics applications. The overall information processing architecture for our robot is shown in Figure 1.

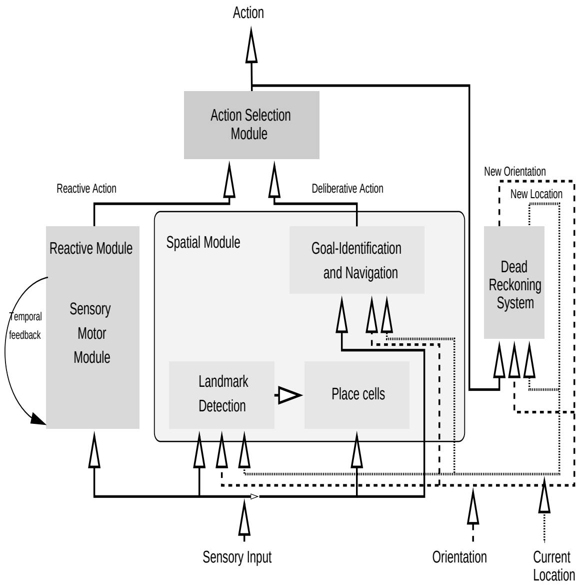

Figure 1: A model of spatial navigation for robots

The model incorporates a dead reckoning system that performs path integration to determine the robot's new location and orientation based on its earlier estimated location and orientation as well as the action performed. This provides a critical input to our place cell system. In addition, our model has two distinct behavioral modules: sensory motor (reactive with possibly temporal feedback) and spatially-deliberative (where particular spatial locations can be chosen as goals). The reactive module is responsible for the normal operation of the robot, including the execution of its intended task. The spatial module is responsible for tracking the movements of the robot and building a spatial map of *interesting* locations encountered by the robot. In times of need, the robot might use its spatial module to determine locations of specific objects and then navigate towards them. In the work reported in this paper, we build in the mechanisms of spatial navigation, i.e., we build in the gross mechanisms of the spatial module. The sensory motor module can be designed either manually or by the process of evolution.

The spatial module is based on a development of place cells as in the case of the hippocampus. We will describe our place cell model later, for the moment it suffices to say that the place cells represent specific portions of the environment with the encoding being a combination of the dead reckoning estimate and the sensory attributes of the cues visible from that location. As the robot moves around, place cells are inducted from a pool of available cells based on novelty of the position and the landmarks visible from that location.

The robot is endowed with a goal identification module which identifies specific spatial locations as goals. In general goals are spatial locations already recorded in the place cells although this is not a necessary requirement in our model. The goal identification process itself may be either autonomous (self- determined by the robot) or programmed by the user (as often happens in robotics). Depending on the desired goal location (and possibly the current sensory state and the state of the place cell system), the spatial module suggests an action for the robot.

The sensory motor module may also suggest an action under the same circumstances. An action selection module decides which of these suggested actions to actually perform. Action selection may be performed in a variety of ways from simple priority based selection to more complex ones involving evaluation of *utilities* of the consequences of the actions.

3.1 Place Cell Representation

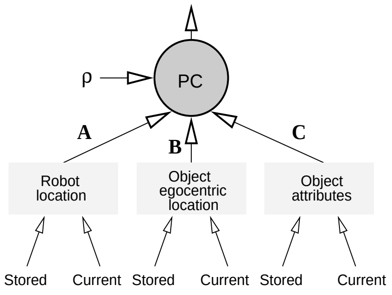

Figure 2: Place cells.

Our place cell model is based on the dead reckoning ability of animals, i.e., we assume that the robot has some way of estimating its current location with respect to some origin of measurement. As with all conventional robots, we assume that our robot is equipped with an on-board compass. We take the start position of the robot as the origin of measurement and thereafter, use dead reckoning to update its location estimate with each move and turn. This location estimate of the robot is one component of our place cell representation, as shown in Figure 2.

We also model the head-direction cells, i.e., we assume that our robot has the ability to determine the heading of landmarks (and other sensory cues) detectable from any given position as well as their distance. This egocentric location of a landmark forms the second component of our place cell system. The final component of our place cell model are the attributes of the sensed object. These attributes might be used to recognize specific objects and distinguish them from others. In addition to these, an adaptable confidence parameter ρ is associated with each place cell and denotes the robot's current level of confidence in that place cell. As we will see shortly, this confers upon the robot the ability to handle dynamic environments and disappearing landmarks. As shown in Figure 2, each place cell has four stored components: ρ, estimate of the robot's location, estimate of egocentric location of sensed object, and finally, the attributes of the sensed object.

3.2 Operation of the Place Cell

At any given point in time, the firing of a place cell is dependent on the three matching functions shown in Figure 2: **A**, **B**, and **C** corresponding to how the stored values of the place cell match the current values available at the robot's current location. For instance, if the current estimate of the robot's location is close enough to (as defined by the matching criterion) the stored estimate, **A** will produce a high value. The other two matching functions operate along similar lines. Our current implementation uses extremely simplified versions of the matching functions, a gaussian match for **A**, and exact matches for **B** and **C**. Since our robot operates in a simulated world, for the moment these serve the purpose. Each place cell operates as per the following algorithm:

Step-1 If **A**, **B**, and **C** are all high, object is recognized again. Increase confidence in place cell: $\rho \leftarrow \rho + \lambda\ (1\text{-}\rho)$. Return ρ.

Step-2 If **A** and **B** are high, while **C** is low, object no longer at that location (must have been removed). Decrease confidence in place cell: $\rho \leftarrow 0.5\ \rho$. Return ρ.

Step-3 If **A** is low but **B** and **C** are high, object recognized but dead reckoning estimate must have drifted. Correct the drift. CurrentRobotLocation ← StoredRobotLocation. Return ρ.

Step-4 If none of the above, return 0.

3.3 Operation of the Spatial Module

In its normal course of operation the robot is driven by its reactive module. During the operations of the robot, the landmark detection system constantly monitors the robot's sensory system for interesting landmarks. If the robot finds an interesting landmark that has not yet been represented in the place cell system (indicated by the absence of place cell firing), it inducts a fresh place cell for that landmark. In addition to initializing the cell with the current robot location, the egocentric location of the landmark, and the landmark attributes, the ρ value is set to 0.75. As was mentioned above, the ρ value associated with a place cell can change depending on the experiences of the robot. It increases if the robot revisits the landmark and the landmark is still there (Step-1) and decreases if the landmark has been *removed entirely* (Step-2). Thus, the ρ value is bounded between 0 and 1. (Note that our model, as it stands, *cannot* correct dead reckoning drifts *and* handle moving objects at the same time.) If the ρ value of a place cell falls below a certain threshold, the cell is decommissioned. This allows the robot to forget useless locations and learn new ones. A common problem in robotics is the drift in dead reckoning estimates. In our place cell model dead reckoning errors can be quickly corrected if the robot visits a location where some place cell was inducted. If the place cell identifies the object (**B** and **C** are high), the current location estimate of the robot can be reinitialized from the stored estimate available in the place cell (Step-3).

4 A Box-Pushing Robot Task

The task we consider here is that of a square arena of 10 × 10 cells, littered with 10 boxes. The robot has to push as many of the boxes to the walls as possible within a limited time. At the end of its simulation trial each box against a wall earns one fitness point and every box pushed into a corner fetches an extra fitness point. In our earlier work we have successfully evolved neurocontrollers for such a robot [2]. In addition to the neurocontroller design, we have also been able to evolve minimal numbers of sensors and their efficient placement [4], leading to interesting designs even in the presence of sensor noise [5]. The task here differs from our earlier work in a few significant ways.

- The robot environment is much larger here, which coupled with the limited sensory abilities of the robot, makes the task of finding and pushing boxes much harder than before. Hence fitnesses observed in these experiments are lower than before.
- Unlike our earlier work, the robot here has a battery with a finite amount of energy, which drains through power consumption of sensors, neurocontroller units, and the robot effectors (primarily the drive train and the wheels).
- The robot operating environment is peppered with power sources, which could be used by the robot to charge up its battery. However, in order to do this, the robot must possess a power source sensor.
- Our earlier sensors could detect boxes, walls, and empty spaces. Here we assume that sensors are tuned to respond to specific classes of objects. For instance, box sensors are only capable of detecting boxes, wall sensors detect walls, power source sensors detect power sources, and internal sensors return a measure of the robot's battery level.
- The robots here have the spatial learning mechanism built in.

The goal of our research was to evolve neurocontrollers for the box-pushing task (corresponding to the reactive module in our information processing architecture). In addition, the sensor types and their placements were also subject to evolution. We wanted to study the effect of incorporating the spatial module mechanism in our robot architecture.

5 Simulation Details and Results

We used genetic algorithms [15] in our experiments with a genetic representation (genotype) of the robot architectures that included the types and placements of the sensors and the input connectivities of the neurocontroller units. This was a slight variation of the representation used in [4]. We created random populations of 500 genotypes. Each genotype was evaluated by decoding it into a robot architecture and placing the robot at a random position in a 10 × 10 grid-world with 10 boxes and 2 power sources located randomly. The robot battery was charged to a maximum of 500 units of energy. Based on the information provided by the sensors, the neurocontroller produces one of four robot actions: move forward, right turn, left turn, and no movement (NOP). If a box happens to be in front of the robot on its forward move, the box (and the robot) move ahead by a square. However, the robot is incapable of pushing two (or more) boxes together and moving through walls. Each sense-act step of the robot takes one unit of time and a simulation trial ends when the robot completes 500 time steps or runs out of battery power. The fitness of the robot is a count of the number of boxes pushed to the walls, with an extra point for every box in a corner. The fitness is averaged over 100 such random environments.

Once the population is evaluated, 100 of the best individuals are selected to be members of the next generation. The remaining 400 population members are produced by choosing parents based on fitness proportionate selection and applying uniform crossover with probability 0.5 and mutation with probability 0.01 per gene. For details, refer to [1]. For each experiment, we made 10 evolutionary runs, each starting with a different random initial population. Since the robots are evaluated over different random environments in each of the runs, it is difficult to directly compare the robots produced by each of the runs. We thus take the best robot at the end of each of the evolutionary runs and reevaluate them on a predetermined set of 500 environments. We refer to this fitness as the *corrected fitness* of the robot [4].

The robots could use upto 5 hidden units in the neurocontroller and upto 10 sensors. Further, the neurocontroller units were constrained to have a maximum of 15 input synapses. The neurocontroller had 4 fixed output units corresponding to the four actions possible (move forward, left turn, right turn, and NOP). All the units compute the bipolar threshold function and the action to be performed by the robot is determined by the output unit that is the winner, i.e., the one that has the highest input activation (refer to [1] for details). The sensors are tuned to detect specific classes of objects located at specific distances from the robot. Further, each sensor can only detect the one grid-cell that it is focused to, and the maximum range of any sensor was limited to 3 cells along either dimension of the grid world. The sensors return a 1 if the object they are tuned to is present in the cell they are focused to, and a 0 otherwise. Power source sensors, when they detect a power source, have the side-effect of charging up the robot's battery by 50 units. Each operating unit of the neurocontroller and each functioning sensor consume one unit of power every time step. Also, the forward move and turn actions require 5 units of power, the NOP's 2, and box pushes, an additional 5 units.

The spatial model is limited to inducting a maximum of 10 place cells, which are only used to encode power source locations. Since power sources can only be detected by power source sensors, robots without such sensors cannot make use of the spatial learning mechanism. The spatial module uses two thresholds: low battery threshold (LBT) and high battery threshold (HBT), which were fixed in some experiments and evolved in others. The goal-identification module constantly monitors the robot's battery level, and if it falls below LBT, it consults the spatial map (place cells) to determine the location of the nearest power source. This is achieved by comparing the current location estimate of the robot with the stored estimates in the place cells. Once the nearest power source location is determined, the robot switches off its sensors and neurocontroller units (since they consume power) and follows a variation of X-Y routing to the goal location. In this process, it navigates by a tactile sensor placed just ahead of it (which is not used otherwise), avoiding obstacles (boxes, walls, and other unlocated power sources) by a random sequence of movements [3]. Once at the goal, the robot charges up until its battery level reaches the HBT. Since in one time step the robot can only charge by 50 units, it might have to spend some amount of time in order to charge to HBT. Time spent navigating to goal locations and charging up there are a part of the simulation time of the robot, hence inordinate time spent in those two activities will leave the robot with less time in which to attend to its box-pushing task. Thus, there is a tradeoff between spending too much time charging up versus pushing boxes, which is what we would like to explore. Once sufficiently charged, the robot switches on its sensors and neurocontroller units and proceeds with its box-pushing task. In the experiments reported in this paper the action selection module is absent; the reactive module is completely in control of the robot as long as its battery level is above LBT, and the spatial module takes control when the battery level falls below LBT.

5.1 Experiment 1: No Spatial Learning

In this experiment, the robots did not have any spatial learning ability (the spatial module was not available). The results over the 10 evolutionary runs are summarized in Table 1. As can be seen, the average fitness is less than one fitness point. The best robot achieved a fitness of 1.48 with a single wall sensor and no hidden units. Most evolved designs do not make use of any power source sensors but almost all of them evolve to have wall sensors. The reason for this might be that these robots do not have the ability to build a spatial map of the power source locations and the only way for them to charge up is by fortuitous contact with the power sources. Since the power sources are randomly located in each trial environment, no particular placement of the power source sensors is particularly beneficial. Apparently, evolution soon discards them. In general, few sensors and few hidden units seem to be favored by evolution in this setting.

	Avg	Min	Max
Fitness	0.994	0.84	1.48
Total sensors	2.4	0	7
Power sensors	0.7	0	4
Hidden units	2.2	0	5

Table 1: Robots without the spatial module.

5.2 Experiment 2: Built-in Spatial Module with Fixed Thresholds

The robots in these experiments were equipped with the spatial learning module. In addition, the thresholds were fixed at: LBT = 200 and HBT = 500. The details of the best robots produced at the end of each of the 10 evolutionary runs are presented in Table 2. The average fitness of the robots is about 52% higher than the ones without spatial modules. The best robot used one wall and two power source sensors, two hidden units, and had a corrected fitness of 2.2. Each of the 10 runs produced robots with at least one power source sensor, thus all the evolved robots utilize their spatial modules. Two of the runs produced robots with one power source sensor and those robots had the least fitnesses of them all. In general, having more power source sensors corresponds to more locations from which the robot can charge up, which combined with the spatial module provides more options to the robot in need of power. Further, the average number of sensors used by these robots was 4, a much higher figure than in experiment 1. Also, the robots that had two or more hidden units (and hence more complex, temporal behavior) had much higher fitnesses (Avg: 2.03) compared to the rest. When the spatial module was lesioned (deactivated) in these robots, the fitness dropped alarmingly (almost 50%). These robots seem to rely heavily on the spatial module to sustain them for longer periods of time, enabling them to move more boxes to the walls.

	Avg	Min	Max
Fitness	1.509	0.95	2.2
Total sensors	4.0	2	7
Power sensors	2.6	1	5
Hidden units	2.2	1	4
Fitness without spatial module	0.76	0.43	1.05

Table 2: Robots with in-built spatial module and fixed thresholds. LBT=200 and HBT=500.

5.3 Experiment 3: Spatial Module with Evolved Thresholds

The thresholds, LBT and HBT are critical to the performance of the robots. A very low LBT may prove detrimental for robots since at the time the robot initiates its navigation to a power source location, it may not have enough power to reach there successfully. On the other hand, a high LBT or a HBT value marginally higher than LBT, will force the robot to keep charging often thereby robbing it of crucial time that could be spent on its box-pushing task. A very high value of HBT will require the robot to waste considerable time at the power source, again leaving it with lesser time to push boxes. In this experiment, we let evolution determine LBT and HBT for each of the robots. The particulars are shown in Table 3. This produces robots with average fitness 34% higher than those in experiment 1 and 12% lower than those in experiment 2. The best evolved robot has a corrected fitness of 2.12 and used 1 wall and 1 power source sensor. It had four hidden units and thresholds: LBT = 198 and HBT = 366. Almost all the runs produced robots with at least one power source sensor. The two designs without any power source sensor had the lowest of the fitnesses. With a deactivated spatial module, the average fitness drops by about 33%, but not as significantly as in experiment 2. Further, even with the spatial module deactivated, the fitnesses of the robots is comparable with the fitnesses obtained in experiment 1, suggesting that the co-evolution of thresholds with the sensors and neurocontroller connectivities produces more robust designs. The robots evolve to use the spatial module effectively while also retaining the ability to perform well in their absence.

	Avg	Min	Max
Fitness	1.329	0.82	2.12
Total sensors	2.7	1	5
Power sensors	1.1	0	3
Hidden units	2.4	0	4
Fitness without spatial module	0.901	0.54	1.22

Table 3: Robots with in-built spatial modules and evolved thresholds.

6 Discussion and Future possibilities

In this paper we have described a spatial model for robot navigation that is largely inspired by neurobiological and cognitive psychology literature. We have realized a simple version of our model and used it in a robot box-pushing task. We have provided the robot with mechanisms for building a spatial map of power source locations and navigating to the nearest one when in need of power. As might be expected, this imparts higher fitnesses to the robots. However, the overall fitnesses of the robots is still very low. Some reasons for this include: the large operating environment, the maximum battery level (which is quite low), the limited sensor model (sensors can only sense objects in one grid-square to which they are focused), inability of the robot to determine when it is stuck in some location or caught in a fixed cycle, and the robot's inability to explore new regions of space. For instance, increasing the maximum battery level to 1000 increased the average fitnesses of the robots of experiments 1, 2, and 3, by about 27% (and a 33% increase for a maximum level of 1500). A number of researchers have observed significant improvements in performance by incorporating noise in their simulation models. With 10% sensor failure probability and a probabilistically weighted action selection mechanism (rather than deterministic, winner-take-all) our robots are able to attain average fitnesses of 3.5 on this task. Experiments are currently underway to explore these issues in detail.

As we mentioned earlier, much of this research is still in progress. We are exploring ways of extending the spatial module to represent spatial locations of other objects (boxes, walls, etc.) as well. With such a mechanism in place, it might be possible for the robots to add an exploratory flavor to their motions. We are also in the process of making the simulation environment more realistic by adding more realistic models of sensing and action. The place cell representation could also be improved. There is evidence that ensembles of place cells code for space [29], rather than single cells as our model has assumed. This is an extension worth pursuing. Instead of X-Y routing, other navigation algorithms capable of operating in non-grid environments would also need to be incorporated. Importantly, the goal-identification module in our experiments, is programmed by the user. We are exploring the use of GP-like constructs for representing such goal-identification programs.

The robots in our experiments did not have any learning ability apart from the spatial learning procedure built in. Also, these robots lacked any kind of an action arbitration mechanism. For the robot to be truly autonomous, it must be capable of identifying and exploiting tradeoffs between its different activities, including actions and goals. For instance, if the robot is operating in an environment in which there are many power sources, then it might choose to run really low on its battery before deciding to charge up. In environments with few power sources, it cannot afford this luxury, and hence might choose to follow a more conservative and cautious approach. In the robots discussed in this paper, this corresponds to some mechanism that could be used to adapt the thresholds LBT and HBT. Similarly, if the robot is pushing a box and finds that it is close to a power source, it might decide to charge up even if its battery level is not really low. The robots might be capable of such behaviors if they had a notion of *utility* and *cost* of performing various tasks, including box-pushing and maintenance of power. We are currently exploring ways of providing our robots with such notions of utility and cost to enable adaptive and autonomous exploration of such behavioral tradeoffs.

References

[1] K. Balakrishnan and V. Honavar. Analysis of neurocontrollers designed by simulated evolution. Technical Report CS TR 95-25, Department of Computer Science, Iowa State University, Ames, IA - 50011, 1995.

[2] K. Balakrishnan and V. Honavar. Analysis of neurocontrollers designed by simulated evolution. In *Proceedings of IEEE International Conference on Neural Networks ICNN'96*, 1996.

[3] K. Balakrishnan and V. Honavar. A neurobiologically inspired model of spatial navigation in robots. Technical Report CS TR 96-19, Department of Computer Science, Iowa State University, Ames, IA - 50011, December 1996.

[4] K. Balakrishnan and V. Honavar. On sensor evolution in robotics. In *Proceedings of Genetic Programming Conference – GP-96*, 1996.

[5] K. Balakrishnan and V. Honavar. Some experiments in the evolutionary synthesis of robotic neurocontrollers. In *WCNN'96: Proceedings of the World Congress on Neural Networks, San Diego*, pages 1035–1040, 1996.

[6] K. Bayse, T. Dean, and J. Vitter. Coping with uncertainty in map learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1989.

[7] R. Brooks. Visual map making for a mobile robot. In *Proceedings of the IEEE Conference on Robotics and Automation*, 1985.

[8] R. Brooks and M. Mataric. Real robots, real learning problems. In J. Connell and S. Mahadevan, editors, *Robot Learning*, chapter 8, pages 193–213. Kluwer Academic Publishers, MA, 1993.

[9] N. Burgess, M. Recce, and J. O'Keefe. A model of hippocampal function. *Neural Networks*, 7(6/7):1065–1081, 1994.

[10] M. Colombetti and M. Dorigo. Learning to control an autonomous robot by distributed genetic algorithms. In J-A. Meyer, H. Roitblat, and S. Wilson, editors, *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, Cambridge, MA, 1992. MIT Press-Bradford Books.

[11] J. Connell and S. Mahadevan, editors. *Robot Learning*. Kluwer Academic Publishers, MA, 1993.

[12] S. Engelson. *Passive Map Learning and Visual Place Recognition.* PhD thesis, Yale University, Department of Computer Science, 1994.

[13] H. Everett. *Sensors for Mobile Robots: Theory and Application.* A. K. Peters Ltd, Wellesley, MA, 1995.

[14] C. Gallistel. *The Organization of Learning.* Bradford-MIT Press, Cambridge, MA, 1990.

[15] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[16] D. Kortenkamp. *Cognitive Maps for Mobile Robots: A Representation for Mapping and Navigation.* PhD thesis, University of Michigan, Electrical Engineering and Computer Science Department, 1993.

[17] B. Kuipers and Y-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, 8, 1991.

[18] L. Kuvayev and R. Sutton. Model-based reinforcement learning with an approximate, learned model. In *Proceedings of the Ninth Yale Workshop on Adaptive and Learning Systems*, pages 101–105, New Haven, CT, 1996. Yale University.

[19] B. Leonard and B. McNaughton. Spatial representation in the rat: Conceptual, behavioral, and neurophysiological perspectives. In R. Kesner and D. Olton, editors, *Neurobiology of Comparative Cognition.* Lawrence Earlbaum Associates, Hillsdale, NJ, 1990.

[20] M. Mataric. Integration of representation into goal-driven behavior-based robots. *IEEE Transactions on Robotics and Automation*, 8(3), 1992.

[21] D. McFarland. *Animal Behavior.* Longman Scientific and Technical, Essex, England, 1993.

[22] B. McNaughton, J. Knierim, and M. Wilson. Vector encoding and the vestibular foundations of spatial cognition: A neurophysiological and computational hypothesis. In M. Gazzaniga, editor, *The Cognitive Neurosciences*, chapter 37. Boston:MIT Press, 1995.

[23] B. McNaughton and L. Nadel. Hebb-marr networks and the neurobiological representation of action in space. In M. Gluck and D. Rumelhart, editors, *Neuroscience and Connectionist Theory*, pages 1–63. Lawrence Earlbaum Associates, Hillsdale, NJ, 1990.

[24] J. O'Keefe and J. Dostrovsky. The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely moving rat. *Brain Research*, 34:171–175, 1971.

[25] J. O'Keefe and L. Nadel. *The Hippocampus as a Cognitive Map.* Oxford:Clarendon Press, 1978.

[26] D. Redish and D. Touretzky. Navigating with landmarks: Computing goal locations from place codes. In K. Ikeuchi and M. Veloso, editors, *Symbolic Visual Learning.* Oxford University Press, 1996.

[27] J. Taube, R. Muller, and J. Ranck. Head direction cells recorded from the postsubiculum in freely moving rats: I. description and quantitative analysis. *Journal of Neuroscience*, 10:420–435, 1990.

[28] J. Taube, R. Muller, and J. Ranck. Head direction cells recorded from the postsubiculum in freely moving rats: Ii. effects of environmental manipulations. *Journal of Neuroscience*, 10:436–447, 1990.

[29] M. Wilson and B. McNaughton. Dynamics of the hippocampal ensemble code for space. *Science*, 261:1055–1058, 1993.

[30] B. Yamauchi and R. Beer. Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man, and Cybernetics - Part B, Special Issue on Robot Learning*, 26, 1996.