Toward Self-Organized Place Recognition

Andrew Stout & Yee Lin Tan
Department of Computer Science
Swarthmore College
{stout,yeelin}@cs.swarthmore.edu

May 19, 2003

Abstract

Nolfi *et al* [12] present a hierarchical method for extracting regularities from a robot's environment using a cascade of prediction networks. However, their system does not integrate motor control, trains and tests on a rigid prespecified circuit, and requires fine-tuning the architecture based on the number of segmentations desired. In this paper we present an alternative architecture, based on that of Blank *et al* [1] which addresses these limitations. The system uses a Simple Recurrent Network to learn a primitive behavior for motor control, while a Self-Organizing Map classifies the hidden state of the network into abstract segments. These segments correspond strongly with useful features in the environment. We verify the viability of the system in a simulated world similar to Nolfi *et al*'s.

1 Introduction

One of the most important building blocks for a developmental cognitive mechanism is the ability to extract from sensory input meaningful regularities, such as similarity or continuity over several time steps, and to cluster these regularities into discrete abstract *segments*. It is the abstraction inherent in the process of segmentation which allows the developing agent to bootstrap from low-level reactive behavior to higher-level intelligence by segmenting large amounts of information into segments tractable for further processing. This segmentation can be spatial, temporal, or a more abstract classification encompassing some notion of the agent's 'state'.

1.1 Place Recognition

In order for a robot—particularly a mobile robot—to act intelligently in an environment, it must understand the "spatial semantics" [7] of its environment: that is, it must recognize and distinguish between different parts of its environment, and it must understand how those different parts relate to one another. That interaction with and understanding of the spatial environment is foundational to many kinds of intelligence is fairly obvious, and has been argued at length by others [7, 8, 2, 3].

Place recognition—the ability to identify a sensor (sonar, laser range-finder, or camera) image as representing a certain spatial location—is one component skill for spatial understanding. It is made difficult by the problem of *perceptual aliasing*: the same place can look different under different conditions or from a different pose, and by *image variability*: different places can look the same, especially with low-resolution sensors such as sonar. The former complication necessitates a method of clustering images which are not similar together by some higher level measure of relation, while the latter requires differentiating between sensor images that look the same through the use of memory.

Place recognition can be understood as a specific instance of segmentation: the task is to recognize the similarity of sensor images (and the other informa-

tion needed to address perceptual aliasing) and to segment or cluster them into different spatial locations.

1.2 Goals

Our big-picture goal in the research described here is to develop a neural robot control architecture with the following qualities:

- It learns with minimal supervision.
- It extracts useful regularities from the environment and classifies similar inputs into meaningful abstractions. What is most important is that these abstractions make sense *to the robot.* It is useful for pragmatic reasons if humans can also make sense of the abstractions, but it is important to avoid anthropomorphic bias (see [1]).
- It covers the full range of the sense-process-act loop; that is, its input is raw sensor information, it processes that information in some way, and its output is raw motor control commands.
- It is hierarchically scalable, meaning that the same basic architecture can be used to construct representations or control behavior at different levels of temporal complexity by cascading the output of one layer or 'instance' of the basic architecture into the input to another.
- It can be extended to implement goal-directed behavior.
- It can be successfully applied to the task of place recognition.

1.3 Overview

In this paper, we address the limitations of Nolfi and Tani's segmentation network [12] for abstracting regularities in an environment by presenting a network architecture that is capable of controlling the robot in addition to abstracting higher order concepts from time-series sensory information. The network architecture we present consists of a Simple Recurrent Network connected to a Self-Organizing Map. Instead of training the network controller on a fixed wall-following behavior, the system we propose is trained on a continuous world exploratory behavior. The system is tested on environments with varying degrees of complexity.

Section 2 discusses Nolfi and Tani's experiment, Blank *et al*'s developmental architecture, and other related work. Section 3 details the design of our system, and the experiments carried out to evaluate the system are presented in section 4. Section 5 synthesizes and discusses the results. In section 6 we discuss several directions we hope to explore in future research, and we draw our conclusions in section 7.

2 Related Work

2.1 Nolfi and Tani (1999) [12]

The experiments described in this paper were initially inspired by [12]. Nolfi and Tani proposed a hierarchical prediction-segmentation architecture for extracting regularities in the environment, and applied their model to the case of a robot navigating in a structured environment. [12] used a three-node recurrent 'segmentation layer' to classify the hidden layer state of a Simple Recurrent Network (SRN)[4]. The SRN's task was to predict the robot's sensor readings at the next time step, while the segmentation layer categorized the SRN's hidden state. The output of the segmentation layer was then input to a second SRN.

Nolfi and Tani trained and tested their architecture on a Khepera robot in a very simple two-room environment, using a hand-coded controller to drive the robot in a circuit along the walls of the environment as it collected and processed sensory data. They report that after training the network had learned to predict the next sensor values to within a reasonable margin of error, and furthermore that the output of the segmentation layer indicated that the network had learned the meaningful distinctions of 'wall', 'corner', and 'corridor'.

While Nolfi and Tani's results demonstrated neural networks' ability to extract temporal and spatial regularities from the environment, there are serious

limitations to their approach which must be overcome in order to scale it to more complicated robotics applications. The first is that their architecture never controls the robot: they explicitly "assume that the behavior of the robot is predetermined and fixed" [12] (p. 7). If the extracted regularities are going to be used to any useful end, the architecture must have a means of controlling the robot's behavior.

Secondly, the environment in which they train and evaluate their network is very structured, and the robot's navigation through it during training is even more structured. Constraining the learning task to one so structured seems to be in opposition to the motivation of constructing controllers capable of learning and representing regularities in the environment. Considering the rigid circuit on which Nolfi and Tani's architecture is trained and tested, and the length of training, it is likely that the network is effectively memorizing the pattern of input, rather than developing any richer sense of the topography of the environment.

Finally, the size of the segmentation layer—and thus the number of segments identified—is prespecified and carefully tuned by a human. For network learning to be domain-general, such tuning and prespecification must be eliminated.

The work described in this paper is a first attempt to address these three limitations.

2.2 Blank, Kumar, and Meeden (2002) [1]

We developed our network architecture independently, but soon realized it was nearly identical to the mechanisms proposed in [1]. Blank *et al* suggest a hierarchical architecture of alternating layers of prediction and abstraction as a general mechanism for developmental robot learning. They cite abstraction and prediction as two fundamental cognitive abilities, and implement these using Self-Organizing Maps [5] and Simple Recurrent Networks [4], respectively.

The work described below adopts a minor variant on Blank *et al*'s proposal and applies it to a task similar to that of [12].

2.3 Other work

Critical of Nolfi and Tani's rather supervised experiment [12], Linåker and Niklasson [9] used a Resource Allocating Vector Quantizer (RAVQ) that extracted higher order concepts from a robot's sensory information as it explored an environment. The design of RAVQ was inspired by adaptive resonance theory (ART) networks that classified information into categories, and had the capability of creating new categories whenever the input information did not closely match existing categories. In the context of localization, RAVQ is a "constructive" world segmenter that dynamically allocates new symbols depending on the complexity of the environment [9]. Tested on a two-room world modeled after that of Nolfi and Tani [12], their RAVQ system segmented the world into three categories corresponding to the symbols 'wall', 'corner', and 'corridor'.

3 Network Architecture & Training

The controller used in all experiments consisted of a Simple Recurrent Network connected to a Self-Organizing Map. The SRN was trained on a hand-coded reactive primitive. A schematic of the system architecture is shown in figure 1, and each component is described in detail in the following sections.

3.1 Simple Recurrent Network

A Simple Recurrent Network[4] predicted the sensor values at the next time step. The network had 18 input nodes, corresponding to 16 sonar sensors and two nodes for the current motor control values (translation and rotation). The input layer was fully connected to a hidden layer of 9 nodes. At each step the hidden node activations were copied to the context layer, which was also fully connected to the hidden layer. The output layer was a prediction of the sensor values and motor control values at the next time step, and thus was identical in size to the input layer.

Similarly to Nolfi *et al* and Blank *et al*, a Simple Recurrent Network was chosen because of SRNs' abil-

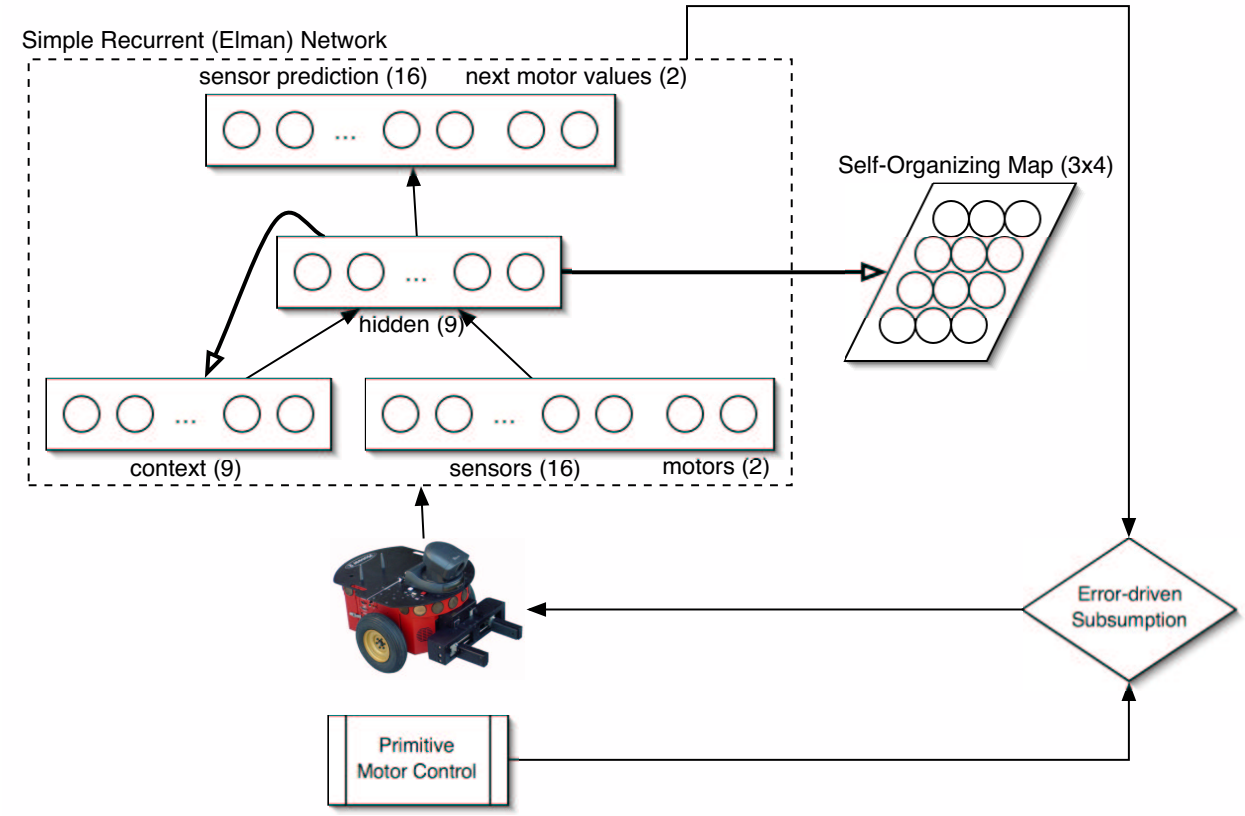

Figure 1: System architecture

ity to make context-sensitive predictions. The context layer allows the previous state of the network to be (recursively) input to the hidden layer, which can then synthesize that input into a kind of 'memory' and make different predictions on the same sensory input based on differing context (that is, previous states). This feature addresses the issue of perceptual aliasing mentioned in section 1, as well as potentially enabling the network to learn multi-step behaviors.

3.2 Self-Organizing Map

The activations of the 9 units of the SRN's hidden layer were used as input to a 3×4 Self-Organizing Map [5], whose target vectors were naturally also nine-dimensional. In the experiments described here the SOM 'output' was simply which node an input mapped to (described by coordinates), because the SOM was a terminal layer in these experiments; however it would be easy to choose a slightly more complicated output description in order to use the SOM output as input to another SRN, as suggested in [1].

As in [1], a Self-Organizing Map (SOM) was chosen as the abstraction mechanism primarily for SOMs' ability to automatically find abstractions without supervision, and because they classify data topologically, allowing relations between data to be easily inferred and easily visualized.

3.3 Primitive Motor Control

The SRN was trained to mimic a primitive motor control behavior, which drove the robot around based on sensor readings. The primitive was a hand-coded, prespecified reactive controller that made decisions based only on the current sensor data. In a developmental sense, one may think of the primitive as innate reflexes, which serve as a foundation for a bootstrapping process to develop more complicated behavior.

We used two primitive behaviors for the experiments described below. Both were very simple reactive controllers. The first avoided obstacles and otherwise followed a straight path. The second was designed to seek open space.

3.4 Learning

We have dubbed the training of the network a "two and a half phase" training:

In the first phase the SRN was trained to mimic the primitive behavior and predict the next sensor readings. Backpropagation of error was used on all connection weights. When error (averaged over 100 times steps) dropped below a threshold, the SRN took over control from the primitive behavior and controlled the robot from its two motor control output nodes.

Once the SRN took control of the robot, the "half" phase of learning began. In this phase the SRN's hidden layer activations were collected and stored. When enough sets of activations had been collected, they were used as initialization data for the SOM.

Once the SOM was initialized, the second full phase of learning was the SOM learning to classify the hidden layer activations of the SRN. The SOM was trained for a prespecified number of steps, after which learning was considered to be complete, and data collection for the experiment was begun.

3.5 Learning Parameters

Epsilon, the learning rate of the SRN, was set to 0.5. The momentum parameter, which indicates the propensity of the network to move in the direction of the path that it has been following in the search space was set to 0.1. The network's tolerance level was set to 0.2. The error threshold, which was the level below which sum squared error (averaged over 100 time steps) had to fall in order to move on to the next phase of training was set to 0.1.

As for the SOM, the initial radius of the training neighborhood was set to 2, while alpha, the learning rate parameter, was initialized to 0.02. Both parameters were set to decrease using an inverse-time function. The Gaussian neighborhood function was used.

4 Experiments

The network architecture described in section 3 was designed for and tested on an ActivMedia Pioneer 2DX mobile robot, which was the reason behind the choice of 16 sonar sensor inputs. All experiments described here were performed in simulation, using Player/Stage [13].

4.1 Environments

Experiments were performed in several different simulated environments, described below.

4.1.1 Basic Environment

The basic environment was modeled after the two-room world used by Nolfi and Tani [12], and consisted of two rooms of different sizes, connected by a short corridor about 1.25m wide, as shown in Figure 2. The room sizes were approximately 5m $\times$ 5m and 2.5m $\times$ 2.5m respectively. Two colored boxes were placed in the middle of the larger room.

4.1.2 Modified Basic Environment

We also modified the basic environment to have a slightly wider corridor between the two rooms. In this environment (shown in figure 3), the corridor was approximately 2m, and both rooms approximately 0.5m longer (in the y direction).

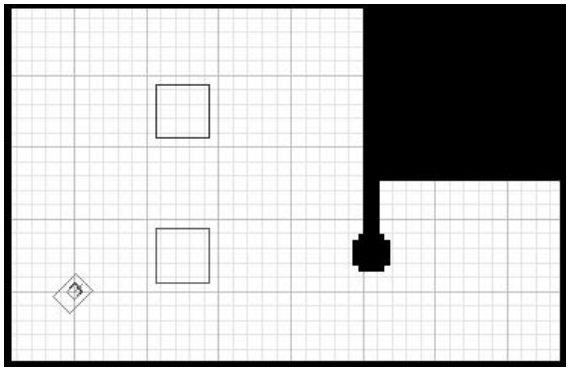

Figure 2: Basic world with the robot at an initial position (1,1) and an angle of 45° with the horizontal.

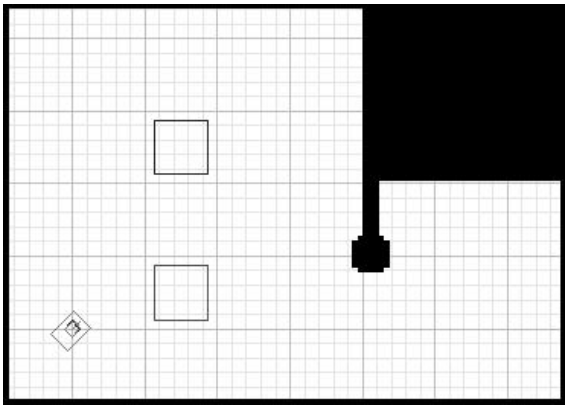

Figure 3: Modified basic world with the robot at an initial position (1,1) and an angle of 45° with the horizontal.

4.1.3 Complex Environment

A third test environment was a section of the floor plan of a hospital. This environment, pictured in figure 4 was much larger (about 11m × 14m, total) and much more complex, with multiple rooms, corridors, and doorways.

4.2 Obstacle-Avoiding Primitive

As mentioned above, two different primitive behaviors were used. In the first set of experiments, the primitive behavior avoided obstacles and otherwise went straight. The network with this primitive was trained and tested in several different trials, detailed below.

4.2.1 Basic Environment

Two trials were run in the basic environment described above: one with a robot started at the bottom left corner of the larger room, at position (1, 1) and an angle of 45° with the horizontal, and the other, at the bottom left corner of the smaller room, at position (6, 1) and an angle of 90° relative to the horizontal.

Training Initiated in Big Room

When started in the big room, the SRN learned to mimic the primitive behavior to within the threshold of error in approximately 14000 time steps. The error plot is shown in figure 5.

After training, the SRN took control of the robot. It successfully avoided obstacles, but since the primitive had "fallen" in to a circuit around the larger room, the SRN followed this circuit as well, without ever entering the smaller room. The trajectory of the robot during the 8000 steps of testing is shown in figure 6.

Through this unforeseen consequence of the property of the primitive behavior, we actually wound up mimicking Nolfi *et al*'s experiment more closely than intended. Fortunately, this did not prevent us from getting encouraging results in the SOM segmentation portion of the experiment .[1] Figure 7 shows a plot of the SOM's segmentations against the world coordinates at which they occurred. Clearly[2] the SOM has segmented the circuit the robot followed during testing into coherent segments. The segments are strongly correlated with features of the world and of the robot's actions at that point. For example, the dark purple portions of the plot in 7 correspond to all of the positions at which the hidden state of the

[1] In fact, it may have helped: it is likely that a robot traveling in a less regular path would have a harder time segmenting what it saw.

[2] or, at least, it would be clear in color...

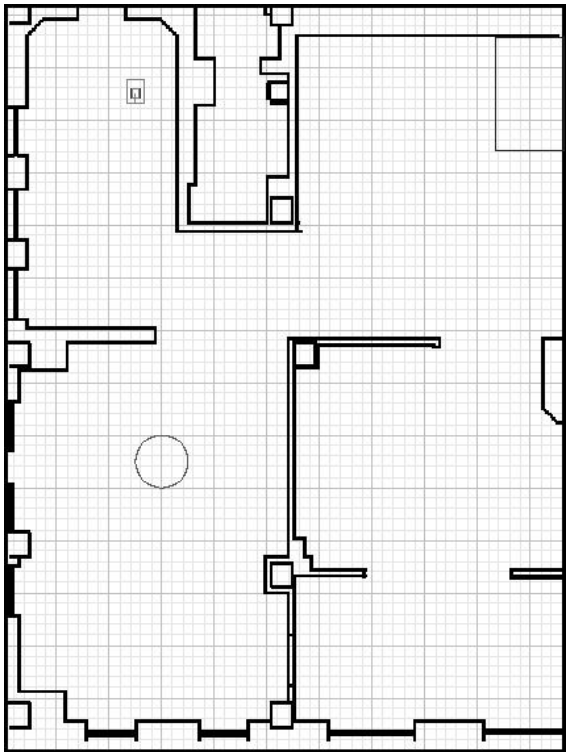

Figure 4: Complex world.

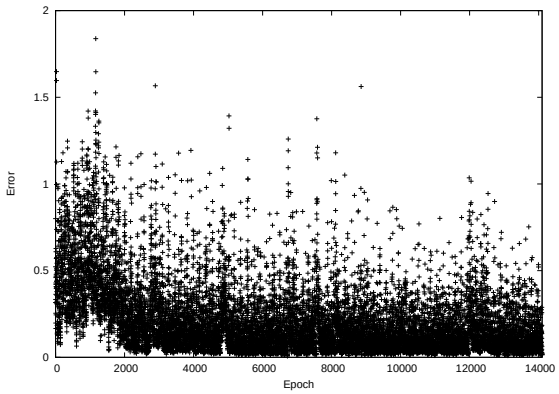

Figure 5: SRN sum squared error while learning the primitive behavior when started in the larger room.

SRN mapped to node (2,0) of the SOM. These locations occur along "straightaways" on the robot's path. Similarly, the light purple sections correspond to the positions at which the hidden state mapped to SOM node (0,3), and these positions all occur in the corners of the circuit (which also correspond to corners of the room). These two segmentations are also plotted individually in 7.

It should also be noted that the SOM segmented two states that only occur in the lower right corner of the circuit, which is different from the rest of the corners of the circuit because it is near the "doorway" to the small room. A plot of the locations of these two states (Y and Z in figure 7) appears in figure 8.

Figure 13 shows the U-matrix visualization of the SOM. The U-matrix was described in the SOM_PAK documentation as a visualization of "the distances between reference vectors of neighboring map units using gray levels" [6]. That is, the darker the gray areas between neighboring nodes, the greater the difference between them. Note, for example, that the distance between C and B is relatively small. Looking back to C (dark purple) and B (dark green) in the segmentation plot (figure 7, top), it is clear that these two states are not very different: both are fairly straight and not near any corners. Contrast this with F and G, which are very different according to the U-matrix visualization. While states F and G occur very close to each other spatially, there is an important difference: in F the robot is still moving fairly straight, while in G the robot is nearing the corner and starting to turn.

Training Initiated in Small Room

When robot training was initiated in the smaller room, the SRN learned to mimic the primitive behavior to within the threshold of error in approximately 9000 time steps. The error plot is shown in figure 10.

The results from this trial were very similar to the trial in which robot training was initiated in the larger room. Like the previous trial, the SRN ended up directing the robot in a circuit around the larger room, never entering the smaller room. Figure 11 shows the trajectory of the SRN-controlled robot during 8000 steps of testing.

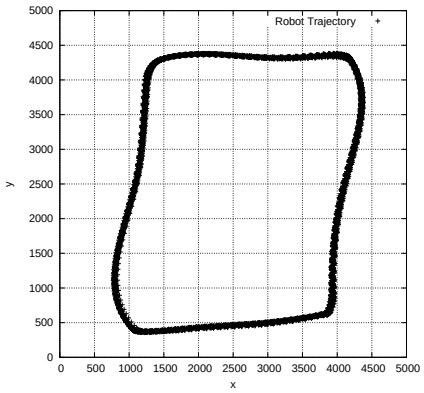

Figure 6: Trajectory of the robot controlled by the trained SRN for 8000 time steps for the trial in which robot training was initiated in the larger room.

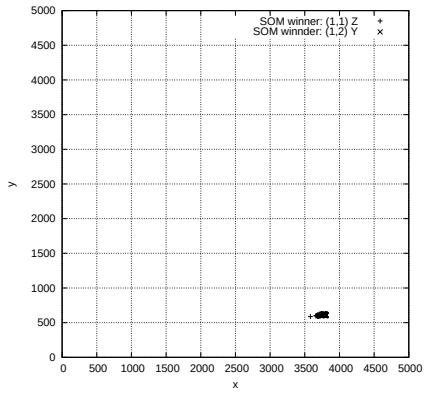

Figure 8: Identification of differentiation: two segments which occur only in the corner near a feature (the doorway connecting the two rooms) differentiating the environment.

The SOM's segmentations against the world coordinates at which they occurred are presented in figure 12. Similar to the previous trial, we observe an obvious correlation between the robot's behavior and the environment's features. For instance, the teal blue portions of the trajectory correspond to all positions at which the hidden state of the SRN mapped to node (2,3) of the SOM. This segment correspond to the part of the robot trajectory immediately after turning three similar corners in the larger room. The lower right corner of the larger room, which is significantly different from the other three corners, was segmented very differently.

Figure 13 shows the U-matrix visualization of the trained SOM. That the sequence of states as the robot traverses the circuit is regular, coherent, and cyclical is reflected in the topological relations of the segments, as shown by the U-matrix. The time-series of SRN hidden states as the robot moves from corner to corner follows a cyclic path around the U-matrix diagram. For example, the cycle ABCDEFGH around the U-matrix corresponds to the portion of the trajectory that begins with teal blue (segment A) at position (1000, 750) and ends in dark purple (segment H) at approximately (1000, 4000). The cycle repeats each time the robot turns a corner.

4.2.2 Modified Environment

After discovering that the network did not learn to explore the small room at all in the above trials, we tried widening the doorway between the two rooms. The network still learned enough to finish training, but regardless of which starting position was used, the SRN controller was prone to crashing the robot into walls.

4.2.3 Complex World

When tested in the complex world, the robot never left the room in which it started, instead developing a circuit similar to those in the basic environment. It completed training and testing successfully, but because it never left the first room, effectively it was not a more complex environment.

4.3 Open Space-Seeking Primitive

The shortcoming of our first primitive prompted us to develop a second primitive behavior which sought open space. This primitive was more exploratory, thus exposing the robot to more of the world. Unfortunately, this controller proved to be much more difficult for the SRN to learn to mimic. With an error

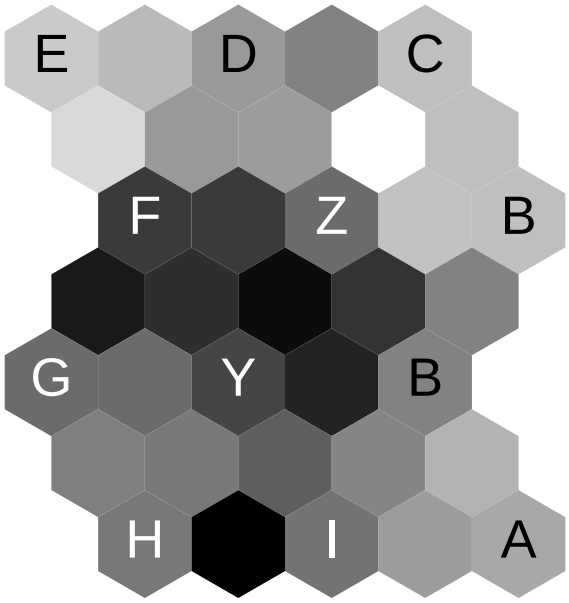

Figure 9: U-matrix visualization of the trained SOM for the trial in which robot training was initiated in the larger room.

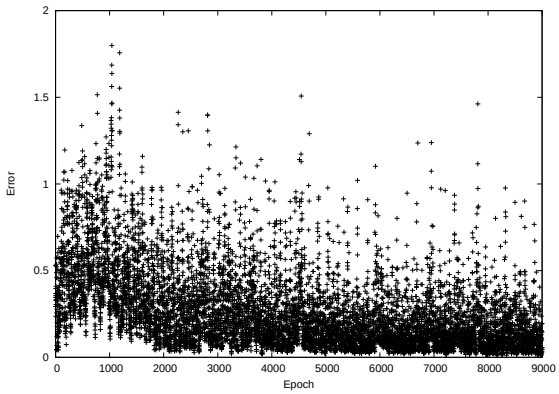

Figure 10: SRN sum squared error while learning the primitive behavior when started in the smaller room.

threshold (the threshold of acceptable average sum squared error to move into the next phase of learning) of 0.1, the network could not converge within 90000 time steps. When the error threshold was raised to 0.25 the network could pass to the next phase of learning, but the SRN controller simply drove in a circle and did not mimic the primitive controller.

4.4 Basic World with Color Blob Vision

In this variant of the experiment, the simulated Pioneer robot was equipped with a blobfinder device that detects color blobs. Only the information from the largest blob was used. The number of input nodes to the SRN was increased to 24, and the hidden nodes, 12. Of the 6 extra input nodes that encoded information returned by the blobfinder, two were binary nodes while the remaining four were real-valued. The binary nodes indicated whether either of the two colors were present. Two of the real-valued nodes provided information on the size of each of the color blobs, while the other two indicated the x position of each blobs' centroid. With this slightly modified experimental setup, we repeated the experiment by positioning the robot in the two different rooms.

Training Initiated in Big Room
In this trial, the network error did not fall below the prespecified error threshold of 0.1. When the error threshold was raised to 0.25, the network managed to complete the learning phase, but nevertheless was unable to mimic the primitive behavior properly. From figure 14, we see that network error did not converge even after approximately 30000 time steps.

Figure 15 shows the trajectory of the SRN-controlled robot over 8000 time steps. The trajectory followed by the robot is not as regular as that of the robot in the basic environment. We also observed that the robot had a tendency to crash into the lower left corner when the trained SRN was in control of the motors. Segmentation of the world was found to be rather poor.

Training Initiated in Small Room
The results from this trial were very similar to the trial in which robot training was initiated in the

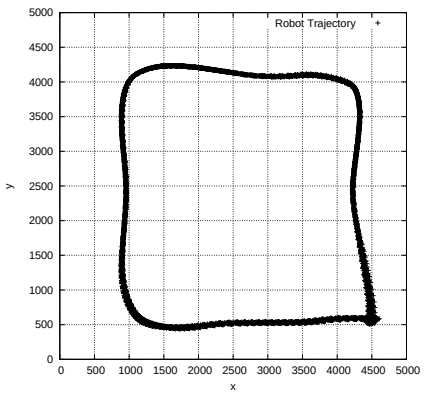

Figure 11: Trajectory of the robot controlled by the trained SRN for 8000 time steps for the trial in which robot training was initiated in the small room.

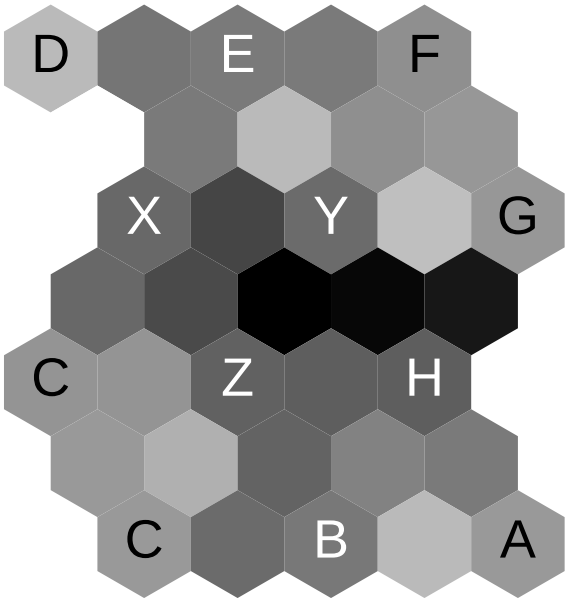

Figure 13: U-matrix visualization of the trained SOM for the trial in which robot training was initiated in the small room.

larger room. Like in the previous trial, the error threshold had to be raised from 0.1 to 0.25 in order for learning to pass to the next phase. Figure 14 shows that the network error over 30000 time steps. The trained SRN mimicked the primitive controller only at three corners. From figure 17, we see that the SRN was prone to directing the robot into the wall at the lower right corner of the environment. As in the trial where training was initiated in the larger room, world segmentation was found to be poor.

5 Discussion

The success of the first experiments described above indicates that our network architecture has the capability to learn the primitive behavior and successfully take over control of the robot. This ability is a crucial criteria for development, and it is encouraging that learned control could be combined with the segmentation task.

However, the primitive behavior learned was too simplistic to result in interesting behavior or effective exploration of the environment, and the SRN was unable to learn the more complex primitive behavior. This is a severe shortcoming which underscores the importance of the primitive behavior, and which must be addressed in order for the proposed approach to artificial cognitive development to be successful. It is possible that the features of a primitive behavior that make it learnable or unlearnable by an SRN can be discovered and articulated, and this would be a great contribution to developmental robotics. The innate instinctual reflexes from which a robot learns are also a prime candidate for evolution, as it has been shown that evolution is good for finding good *starting conditions* for learning successful solutions [11].

Color blob information did not help the robot to segment the world better, as intended. Instead, the extra information that the color blobs provided seemed to have made it more difficult for the SRN to learn the primitive behavior. We believe that this is a consequence of the fact that the information provided by the color blobs were not used by the primitive controller in determining the appropriate motor values given current sensory information.

The first experiments also prove that a Self-

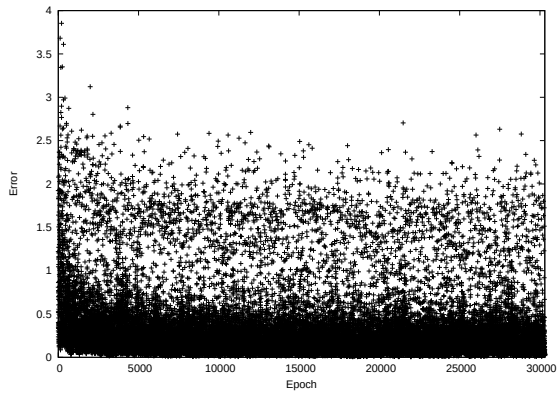

Figure 14: SRN sum squared error while learning the primitive behavior when started in the big room with color blob vision.

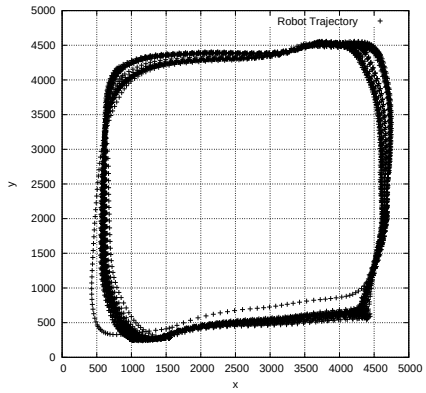

Figure 15: Trajectory of the robot controlled by the trained SRN for 8000 time steps for the trial in which the robot utilized color blob vision and training was initiated in the larger room.

Organizing Map can be used to coherently segment the robot's state into meaningful segments. The segments classify the state of the robot according to a notion of state encompassing both perception and action. The segments also indicate the capability of the network to identify unique places in the environment by segments which occur only in that place.

The architecture described above does not address the issue of perceptual aliasing: the four similar walls and three similar corners were mapped to the same segments as each other, showing that the network did not learn to classify the similar corners differently based on memory. However, it should be noted that in [12] this level of segmentation was not achieved until a second prediction-segmentation layer was added. We are optimistic that our mechanism could achieve equal if not better performance with a second level of hierarchy. Because the primitive forced the training into a circuit, it could not be tested whether our mechanism resolves image variability. We suspect that such classification actually requires a fairly advanced understanding of the "spatial semantics" of an environment, and that it will be difficult to achieve in a small number of levels of hierarchy.

Using a SOM for segmentation in fact had the same complication that [12] briefly mentioned concerning their architecture: the SOM uses almost all of its available nodes to signify a coherent segment, with the result that segments we humans understand as coherent ("corner") get broken into multiple segments by the SOM. Here we must be careful not to unduly criticize the segmentation merely because we do not understand it— doing so would be a prime example of the anthropomorphic bias which [1] cautions us to avoid. However, there is some danger that insufficient abstraction is taking place, because in this case the SOM is given the option of more possible segments than is really needed. A better solution would be some mechanism which automatically determines the correct number of segmentations for 'optimal' classification. Such solutions are notoriously difficult to come by, but [9] is a promising method.

Another potential shortcoming of the architecture advanced here—at least for the purpose of place recognition—is that the network's state is an inseparable combination of perceived state of the world, memory of context, and planned action, and thus a place cannot be recognized independent of the action the robot is taking at that place. It is not entirely clear how those would be separated, but it is clear that significant alterations in the network architecture would be necessary.

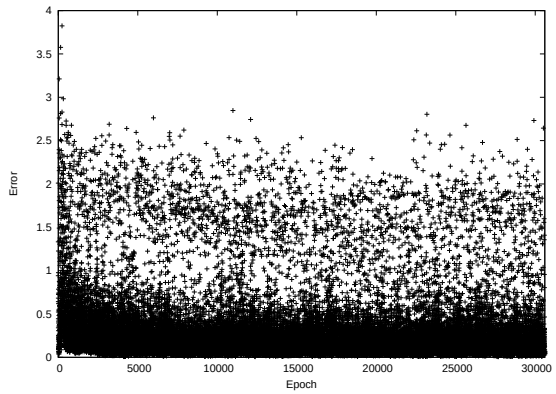

Figure 16: SRN sum squared error while learning the primitive behavior when started in the small room with color blob vision.

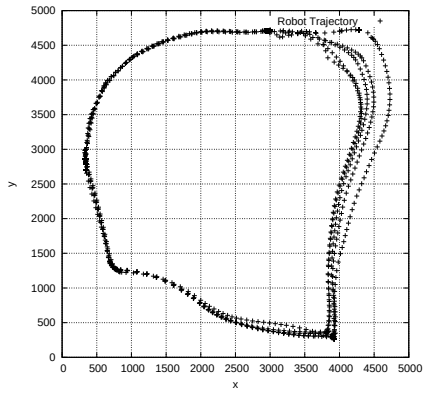

Figure 17: Trajectory of the robot controlled by the trained SRN for 8000 time steps for the trial in which the robot utilized color blob vision and training was initiated in the smaller room.

6 Future Work

The successes and challenges described in this paper suggest numerous directions for future research, many of which we hope to pursue.

The topological coherence of the SOM's segmentations, as illustrated by the U-matrix diagram in figure 13, bodes well for the possibility of integrating a mechanism for goal-directed behavior into the network architecture. The basic idea is to coax the state of the network—and thus the robot—from the current state (marked S in figure 18) to the goal state (marked G in figure 18) by replacing its hidden activation with the target vectors of the segments along the path from start to goal, as illustrated in figure 18. We hope to pursue this idea in the near future.

Difficulty learning the primitive behavior was a serious limitation in the later experiments described in this paper. In addition to finding effective primitive behaviors which are easy to learn, either by trial and error or through a more formal process such as evolution, there are at least two other things we hope to try to improve the performance of the SRN. The first is to abstract the sensory input using a SOM before inputting it to the SRN. This is the approach Blank *et al* take in [1], and it may simplify the learning task. The second possibility is to try Nolfi's idea of emergent modularity [10] in order to learn modular primitive behaviors.

For truly complex behavior, low-resolution range sensors will not be sufficient. Eventually we must successfully integrate vision into our system. This will require both image processing techniques external to the network, and potentially also changes to the network architecture itself in order to accommodate visual information.

[12] also tested the robot on its ability to recognize change in the environment, as indicated by an increase in prediction error. We have not tried this, but if it were clear how such a detection could be integrated back into the system in order to effect behavior, we would certainly follow that line of inquiry.

While simulation is convenient and useful for proof-of-concept experiments like the ones described in this paper, there is no substitute for testing on a real robot in the real world. We have access to a Pioneer 2DX, and hope to develop real-world tasks to evaluate the mechanisms described here.

While our architecture was designed with the explicit intent of hierarchical cascading as a means of development, the scalability of this approach is still largely speculative. As the mechanisms for abstraction and learning are perfected, it will become im-

portant to test the scalability of the architecture to more complex tasks and richer perception.

7 Conclusion

In this paper we have presented an attempt to address limitations in Nolfi *et al*'s architecture for extracting regularities from the environment and in their methods for training and testing that architecture. We present an architecture, based on the abstraction and prediction hierarchy for development presented in [1], which integrates motor control into the learning system and partially addresses the limitations of a pre-specified number of segmentations, and performs over a continuous range of motion, rather than a single predetermined track. We verified this architecture in experiments performed in simulation in which a Simple Recurrent Network learned to mimic an innate primitive motor controller, and a Self-Organizing Map segmented the SRN's hidden states into coherent segments corresponding to the robot's spacial location and behavior. Analysis of the topological structure of the SOM further suggests mechanisms for incorporating goal-directed behavior into the system. Finally, development of higher-level segmentations and behaviors could be achieved by cascading alternate layers of abstraction and prediction.

References

[1] Douglas Blank, Deepak Kumar, and Lisa Meeden. Bringing up robot: Fundamental mechanisms for creating a self-motivated, self-organizing architecture. In workshop on *Growing up artifacts that live* at *Simulation of Adaptive Behavior*, 2002.

[2] Rodney A. Brooks. Intelligence without reason. In John Myopoulos and Ray Reiter, editors, *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 569–595, Sydney, Australia, 1991. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

[3] Rodney A. Brooks. Intelligence without representation. Number 47 in Artificial Intelligence, pages 139–159. 1991.

[4] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[5] Teuvo Kohonen. *Self-Organizing Maps*. Springer, 3rd edition, 2001.

[6] Teuvo Kohonen, Jussi Hynninen, Jari Kangas, and Jorma Laaksonen. Som_pak: The self-organizing map program package, 1995.

[7] Benjamin Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.

[8] George Lakoff and Mark Johnson. *Metaphors We Live By*. University of Chicago Press, 1980.

[9] Fredrik Linåker and Lars Niklasson. Sensory flow segmentation using a resource allocating vector quantizer.

[10] Stefano Nolfi. Using emergent modularity to develop control systems for mobile robots. *Adaptive Behavior*, 5(3-4):343–364, 1997.

[11] Stefano Nolfi and Dario Floreano. Learning and evolution. *Autonomous Robots*, 7(1):98–113, 1999.

[12] Stefano Nolfi and Jun Tani. Extracting regularities in space and time through a cascade of prediction networks: The case of a mobile robot navigating in a structured environment. *Connection Science*, 11(2):125–148, 1999.

[13] Richard Vaughan, Andrew Howard, and Brian Gerkey. *Player* and *Stage*. University of Southern California Robotics Laboratory & HRL Laboratories Information Sciences Laboratory, 1.3.1 edition, 2002.

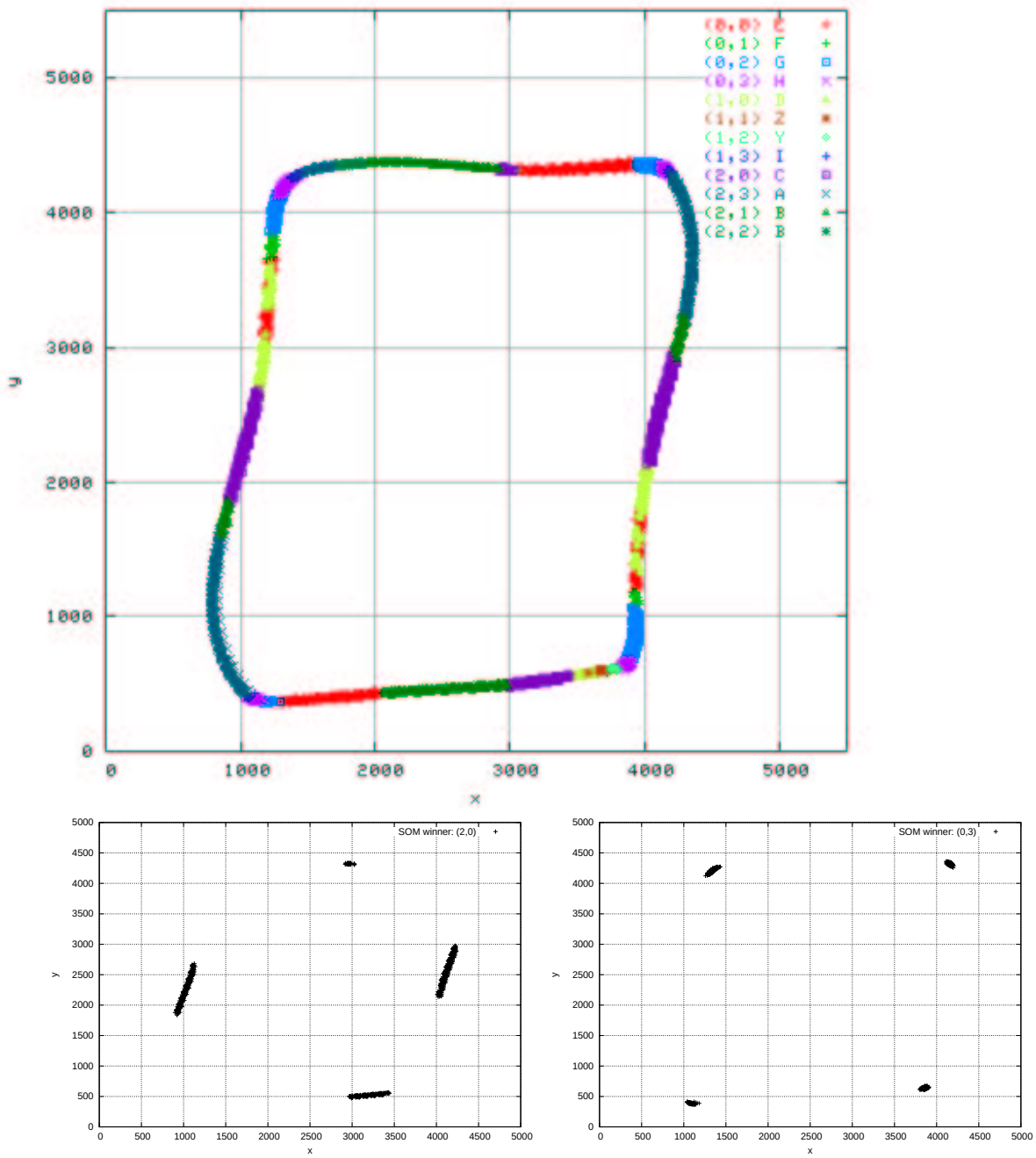

Figure 7: TOP: SOM segmentation of the SRN's hidden state plotted at the location at which each segmentation occurs. The segmentations are strongly coherent and strongly correlated with the robot's location and behavior. The grid in the figure corresponds to the grid in the picture of the environment in figure 2. LEFT: The locations of those states that mapped to node (2,0) in the SOM. RIGHT: The locations of those states that mapped to node (0,3) in the SOM. These plots correspond to the trial in which robot training was initiated in the larger room.

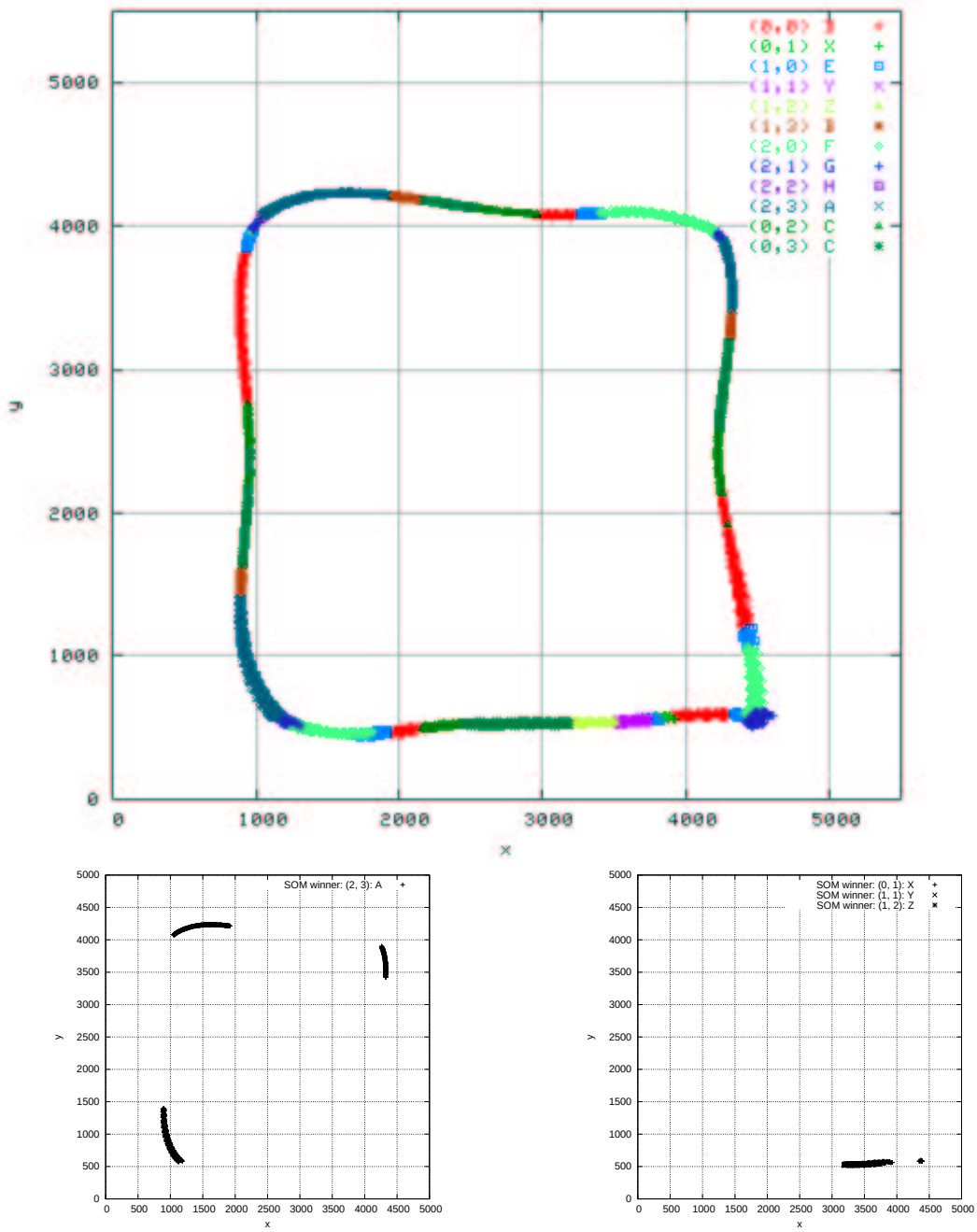

Figure 12: TOP: SOM segmentation of the SRN's hidden state plotted at the location at which each segmentation occurs. LEFT: The locations of those states that mapped to node (2,3) in the SOM. This segment occurs in three similar corners in the environment. RIGHT: The locations of those states that mapped to nodes (0,0), (1,1), (1,2) in the SOM. These three segments occur only in the corner near the doorway connecting the two rooms. All of these plots correspond to the trial in which robot training was initiated in the smaller room.

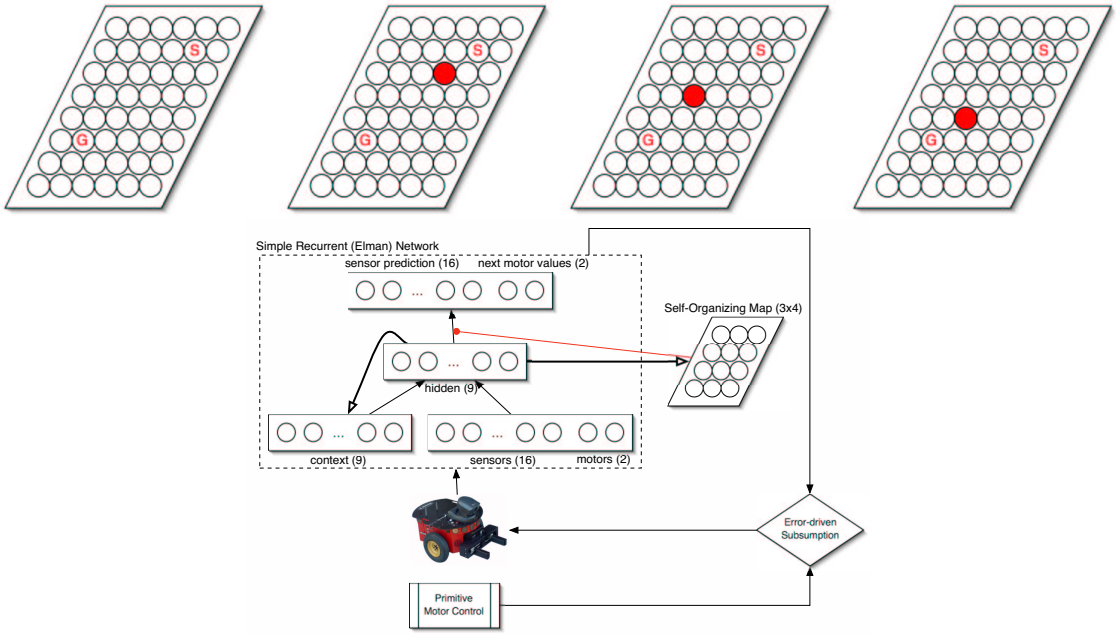

Figure 18: Goal-directed SOM activation. By replacing the SRN's hidden state with the target vectors of SOM nodes on a path from the current state S to the goal state G, goal-directed control may be possible with only slight modifications to the current network architecture.