Symbol Self-Localization in Mobile Autonomous Robots Prediction Networks and Self-Organizing Maps

Eric Levy

Spring 2003

Abstract

Symbol localization partitions a world into a set of meaningful symbols and identifies current position in terms of that set. High-order localization resolves ambiguities in current sensory to uniquely identify a robot's location in a world by combining a long-term memory of the world's structure with a short term memory of recent action and sensory. A hierarchical architecture of developmental components, similar to Nolfi et al.'s [4], comprising a Self-Organizing Map of hidden-layer activations of a Simple-Recurrent Network prediction sensory state is able to develop a meaningful set of symbols with some high-order properties, and to localize in terms of that set, but not able to develop a clean, consistent, high-order set of symbols, or to develop a set of symbols well-aligned with that a human might produce. To remedy this shortcoming, we propose adding additional components, such as a second SRN and a second SOM, to perform the higher-order symbolization and localization.

1 Introduction

1.1 Importance of Localization

Navigation is paramount for any mobile autonomous robot to perform even the simplest tasks. For those tasks, it must merely detect the inhomogeneities of its visible surroundings and set its current course accordingly. In that case, it is sufficient to understand a robot's surroundings as space and time occupied by objects of various sorts that move into, within, and out of a its sensory. Navigation is then merely a constant readjustment of course calculated from current sensory state.

For instance, a robot charged with the simple task of collecting trash until it detects an obstacle, which it will engage or avoid depending its determination of it to either a target or otherwise, may succeed simply by wandering so as to avoid non-target obstacles until it identifies a target within its sensory.

Navigation by detection and reaction to spatial and temporal inhomogeneity is however only a trivial example. As a task grows in complexity it demands that the robot be considered a component of a world with unique features in time and space that can be generalized into a world understanding. Returning to our trash collection task, we may wish a robot to move collected trash to a specific position in the world, even though we have not placed a waste receptacle in that place. Because this dumping station is not explicitly marked, the robot must perform a more advanced variety of navigation than the trivial case of mere collection in which it reaches the target location by using the detectable features of the world to pinpoint, or at least guess, a path to that destination.

In order for the robot to reach its destination, it must apply knowledge of its position to plan and follow a path, and it must continually correct that path to account for natural, random deviations from course. To identify that path, it must localize itself in space and time within its world. Detection of nearby features of the terrain and determination of its possible locations, called localization, is clearly indispensible to this more complex navigation, that is, navigation when we from the outside impose the uniqueness of certain regions.

We thus require an architecture for a robot brain to perform the self-localization of mobile autonomous robots. We currently consider the localization problem exclusively, but with the idea that as solutions to it emerge, we and others may study their application to navigation. The most obvious technique is to analyze currently visible landmarks. One can imagine how a set or range sensors allow us to determine whether the robot is beside a corner or beside a straight wall. However, this does not uniquely identify robot's position in most worlds, which have multiple corners, multiple straight walls, and so forth. In some cases, range sensors on the other side of the robot would help to resolve this ambiguity, but generally the far wall will be out of range. In general, current sensory state is insufficient to the complete the localization. Later, we discuss experimental support for this speculation.

1.2 Importance of Time-Dependent Localization

The problem of the ambiguity in location from sensory state is much the same as that of us navigating on the roads. We can classify our current road as residential or commercial, local or highway. Yet we cannot identify its name, for often, even when the roads are vaguely familiar, different roads or sections of the same road will be indistinguishable. The houses or businesses will all share the same general character, and our memories will not be detailed enough to remember features of individual buildings. Our solution is to remember our current road and the approximate time we have been on it. Combining our recent memories with our knowledge of our desired route, we are able to accurately determine our current localization. A robot able to self-localize is equally dependent on temporal data. We thus demand that our localization architecture consider not only the current sensory state but also those of both the recent and distant pasts.

During navigation both we and robots perform two orders of abstraction. Symbols determinable with only current sensory state is *low-order*, those into ones with sensory state and memory are *high-order*. Low-order symbols contain high-order ones on an organizational hierarchy, since high-order ones apply additional information to add distinction to low-order ones. Also, although time-independent localization is capable of only low-order symbol localization, time-dependent localization is capable of both orders.

1.3 World-Independent and Teacher-Independent Localization

It would be impressive to develop an architecture that, after being provided with a map of its world, could succeed in localization. However, we believe it is most impressive to work toward an architecture within the framework of developmental robotics. Such an architecture would first learn the key elements of an arbitrary world and then report its probable location or a set of its probable locations as it moves. Although neither the past studies nor ours are tested for world independence, they apply the developmental techniques with this eventual goal in mind.

Along the similar lines, we hope for a system functional without a teacher. That is, we want a robot to independently develop an understanding of the world sensible to it. Although a robot taught to recognize symbols taught from the outside, could be useful, it is not the area of our interest.

1.4 Symbol Localization Problem

As we move through the world, we consider our positions not in terms of a coordinate system, but rather in terms of regions determined by our functional abstraction of the structure of the world. With these regions we associate symbols that enable us to communicate these regions and to record them. For instance, we consider our location within an apartment as either in the kitchen, in the bathroom, or in the bedroom. In this case, the regions and their function is blatant. But even in a world with less clear boundaries, we are usually able to apply this symbol partitioning. On a university campus, we may specify an outdoor location by noting the set of surrounding buildings. If that location is special because in has benches and gardens, then we will give that region a proper name and categorize it as a park.

Earlier in our trash collection and dumping task, we considered a robot that would navigate to a position we specified. But if a robot brain architecture is able only to localize in terms of position coordinates, and we wanted it to move trash to a specified room in a building, we would have to give that room in terms of its coordinates. If we want robots to interact with us in a way with which we are comfortable, we demand that they themselves distinguish and identify individual rooms. For robots to perform tasks useful to us in the world in which we live and work, they must partition the world as we do.

The general problem of partitioning a space into a comparatively small set of higher order abstractions is called the Symbol Grounding Problem. Our problem to map recent and current sensory into a set of meaningful symbols is an instance of the Symbol Ground Problem we call the Symbol Localization Problem. We explore several time-dependent and -independent architectures for symbol localization, analyze the strengths and weaknesses of each, and, in light of these observations propose and test our own architecture for localization. However, it is first necessary to introduce an experimental paradigm we call the *Two Room World* used in ours and several others' experiments.

2 An Experimental Paradigm: Nolfi and Tani's Two Room World

Nolfi and Tani have developed a world for mobile robot simulations for testing a robot's ability to learn to self localize. Thus, it is ideally suited for tests of time-dependent localization [4]. It has a variety of local features, including corners, straight walls, and a corridor. It comprises two rooms, a large and a small, connected by short corridor. And although it has no symmetry, so every position is unique; because it is a larger scale than that of the robot's range sensors, the robot cannot distinguish among the various corners or straight walls using current sensory information. Our reproduction of the world appears in Figure 1. Throughout, we call this world the *Two Room World*.

In the experiments involving the Two Room World a robot navigates a counter-clockwise wall-following direct control. This control algorithm is sensible because although its implementation is simple, it forces the robot to experience all the features of the world in regular patterns. The control is direct to create regular periodic pattern from which an architecture may learn temporal and spatial relations to resolve the ambiguity of sensory information. To navigate, the robot uses eight range sensors restricted to a distance smaller than the length of the big room.

For controlled comparison with the work of Nolfi and Tani and of Kröse and Eecen [2], we also apply the Two Room World in our experiments.

3 Previous Work

Localization is an important problem already studied extensively within the developmental paradigm. One technique, the most obvious, is to determine position relative to visible landmarks of known and stable positions. Indeed, Kröse and Eecen apply this technique successfully for an asymmetric world of the same scale as the sensory range of the robot. However, limited sensory range and symmetry clearly lead to degeneracy in the mapping between sensory and position, so that this technique is not generally viable. Lanåker and Niklassen also apply this technique in the two room world, but again, they fail to resolve sensory ambiguities. However, Nolfi et al., the inventors of the two room world, present a hierarchal architecture that succeeds in that task by separating high- and lower-order abstraction into separate components.

3.1 Time-Independent Symbol Localization

3.1.1 Self-Organizing Map Variant for Organizing Sensory Information

Kröse and Eecen applied a variation of the Self-Organizing Map (see Section 4.1) to associate cur-

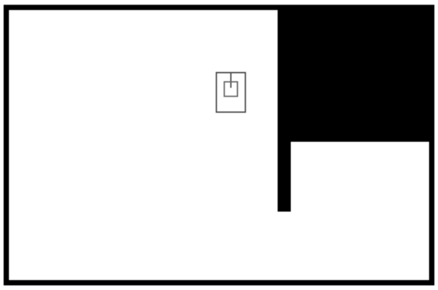

Figure 1: The world originally developed by Nolfi et al., which we call the *Two Room World.* It comprises several unique features of terrain, including corners, straight walls, and a corridor. However, a robot cannot distinguish among the various corners and walls by considering only the current sensory states.

rent sensory with position and orientation to perform localization and navigation. Their architecture mapped position space into a finite number of symbols, so that, once trained, it could report its current region and the nearby ones.

They tested the ability of their architecture to self-localize on two distinct worlds. Both were deliberately designed to have no symmetries. The first was rectangular except that two adjacent corners had blocks of different dimensions, so that they were uniquely identifiable from range sensor readings. The second world thes same except that it had in addition large blocks along two of the walls. Both worlds were of the same scale as the sensory range of the robot.

Localization succeeded in the first world but failed in the second. The success in the first world hinged on the robot's general ability to detect its relative position to one of the distinguishable corners In the second world, however, the large blocks occluded the distinctive corners from the range sensors and were themselves too large to be uniquely identifiable. Thus, localization and navigation were not possible in that world. Kröse et al.'s results clearly demonstrate that their architecture is extremely fragile and not useful in most worlds or on most robots. Certainly, it would not be able to localize itself within the two room world, where three corners and three walls appear identical on the local scale. Rather, it would only be able to determine the nearby features of the world, that is, whether it is near a corner or near a straight wall.

3.1.2 Resource Allocating Vector Quantizer for Segmentation

Lanåker and Niklassen performed simulations of a Khepera robot in the Two Room World in which they used current sensory state input as input to a Resource Allocating Vector Quantizer (RAVQ).[3] A RAVQ is an architecture for un-supervised learning that maps each input vector of an arbitrarily dimensioned space to one node from a set. Notice that the domain and codomain of this algorithm, are the same as that for the Symbol Grounding Problem, so the RAQV may be ideally suited to immediately solve the problem. However, whether the RAQV will abstract the symbols we want is an empirical question Lanåker et al. seek to answer.

Lanåker et al. detail the mathematic derivation of the RAQV, but we provide only a qualitative summary sufficient for our analysis. The distinctive feature of the the RAVQ is that the number of output nodes is dynamically determined by training input. The RAVQ is initialized with a single node and parameters for determining the greatest dissimilarity possible of different input vectors mapped to the same node. During training, the algorithm adds new nodes. To each node is assigned a model vector which must be sufficiently similar to input vector mapped to that node. When the training algorithm encounters sufficiently many inputs sufficiently distinct from available model vectors, it adds a new node with a model vector satisfying that input. Thus, the RAQV partitions a vector space into a number of regions sensible given the variation of inputs, and dependent on the parameters. More, when it encounters input of novel character, if long after initial training, it promptly adds new nodes as necessary.

The goals and successes of Kröse et al. were the immediate partitioning of robot sensory motor state and the immediate response to changes in the Two Room World. Performing the wall following behavior characteristic of these experiments, the RAQV identified three symbols: corner, straight wall, and corridor. This is much the same result one would expect from a human asked to identify the primary features of the world.

Kröse et al. realize that the real world is dynamic. It has people and robots, which locomote and which move, build, and destroy objects and terrain. It not the focus of our discussion, but it is nonetheless true that if we wish to solve the general problem of symbol localization, we must build architectures able to properly respond to by altering their world knowledge throughout their instantiation.

Kröse et al. performed an second experiment in which, after the RAQV was trained in the two room world, they added the an inner wall to one of the corners, so that the corner was also a corridor. Before the robot finished negotiating the corner corridor, the RAQV identified this state with a new, fourth node. The RAQV once again acted much like we might expect from a human, this time responding a novel category of sensory induced by a change in the world.

Despite these successes, these studies have not solved the Symbol Localization Problem because, for the same reasons we already identified, this time-independent approach cannot resolve the ambiguities inherent in current sensory. However, Kröse et al. have partitioned current sensory state into a set of meaningful symbols and have done so dynamically. Symbols that consider only current sensory state we call *lower-order symbols*. We now turn our attention to work toward time-dependent symbol localization and ask whether we can succeed in the symbol localization.

3.2 Time-Dependent Symbol Localization

The least subtle approach to time dependent localization would be to use as input to successful techniques in the time-independent case, a record of previous sensory information However, one or even several previous time steps probably will not suffice to capture the relevant navigational history of the robot. The robot may have been navigating a similar part of the world for an arbitrarily long time, so that very recent sensory history adds no information to current readings. Increasing the length of the record adds no guarantee that it will contain relevant information, and is impractical because of constraints on memory and on computation. Clearly, considering a simple record of past sensory is not a viable solution.

We might next suggest building a record of only distinct sensor readings. Then, redundant information would be discarded. We might improve the usefulness of this record by adding the length of each period of relative constancy. Unfortunately, we lack a sound method for determining whether two sensory states are sufficiently different to merit a record of change. Euclidean distance in sensor space, for instance, is certainly a poor indicator of such difference because an important change in the world may be apparent to only to only a few out of many sensors.

We find no sound approach for applying sensory information directly to a high-order abstraction architecture. The primitive preprocessing of current and past sensory state before input into a developmental architecture will not provide useful data for symbol localization. Instead of preprocessing, we must apply another developmental architecture prior to symbol localization. That architecture must generalize the current and past sensory state so that it can be represented in a small amount of meaningful data. This results in a hierarchy of developmental architectural components.

3.3 Hierarchy of Simple Recurrent and Segmentation Networks

Nolfi and Tani have developed and tested a hierarchical architecture that can perform rudimentary teacher-dependent symbol localization. The system runs in phases such that lower components in the hierarchy are trained in early phases. Their system has important drawbacks, foremost among them that the number of symbols must be determined in advance to training. However, it provides a basis and results important to our experimental work.

Like Lanåker et al., Nolfi et al. performed wall-following direct control on a simulated Khepera robot in the Two Room World. They used a three-level hierarchical architecture of developmental components to successfully symbol self-localize. At the bottom, they trained a Simple Recurrent Network to predict future sensory state. They then used the hidden-layer activations of that network to as inputs to a Segmentation Network, a specialized network of unsupervised learning with three output nodes, three input nodes, and no hidden layers. Finally they train a second SRN to predict the next change in the Segmentation Network winner and the time interval for which it will remain the winner. They performed various experiments on several variations of this architecture. We discuss those most relevant to our problem of self-localization.

3.3.1 Level 1: Simple Recurrent Network

During the first phase of the experiments, Nolfi et al. trained a Simple Recurrent Network to predict the next sensory state. The input layer comprised ten nodes, eight given normalized readings of the eight Khepera sensors and two given normalized representations of the velocity of the two Khepera motors. The output layer comprised eight nodes given normalized readings of the sensors in the next time step. (Of course, Nolfi et al. and we really mean input comprised information from the last time step and the output information from the current. We adopt this convention throughout.) The hidden layer comprised three nodes. The learning rate was $\epsilon = 2.0$ and the momentum was 0.0. The network was trained as such for 100,000 time steps.

A simple recurrent network of this design is sensible for two reasons. First, as with an Auto Associative Network, the output of this network is sufficiently determined by hidden-layer activations, the comparatively small hidden layer forces the encoding of data vectors into vectors of lower dimensionality, and thus of greater abstraction. Thus, this SRN learns abstract representations of data. Second, as does any Recurrent Network, this network maintains a memory because it receives as input the previous hidden-layer activations in addition to data from the input layer. In this way, the hidden-layer activations come to encode not only the current sensory, but also an abstraction of the networks history. Thus, time-dependence is an inherent part of this Simple Recurrent Network.

Nolfi et al. found that the sum-squared network error decreased consistently throughout training, which suggests that the hidden-layer activations were converging on a time-dependent abstraction of sensory-motor state.

3.3.2 Level 2: Segmentation Network

During the second phase of the experiments, Nolfi et al. trained a specialized network called a Segmentation Network. They provide its mathematic definition, but we discuss only its important qualitative features. The Segmentation Network has an input layer, an output layer, no hidden layers. It conducts time-recurrent un-supervised learning that places each input vector into a category determined by a winner-takes-all design. In this case, the segmentation network had three input nodes, and three or four output nodes. The inputs were the hidden-layer activations of the fully-trained Simple Recurrent Network. The Segmentation Network was trained for 100,000 steps.

When they compared the winner of the Segmentation Network with the three output nodes to the absolute position of the robot, they found that the learned set of three symbols represented the same set of states across ten experiments. One symbol represented navigation along a straight wall, one around a corner, and one in the corridor. In certain ways, this result is remarkable, because this segmentation is exactly that that we before said a human would likely choose and that Kröse and Eecen's RAQV was shown to choose.

When they repeated with four output nodes, they observed results inconsistent across experiment and decreasingly stable with time.

3.3.3 Level 3: Upper Simple Recurrent Network

The highest component of the hierarchical architecture was a second SRN, the Upper SRN. The input layer comprised four nodes, three given the current winner-takes-all output of the Segmentation Network and the fourth given the number of time steps it has been in that state. The output layer comprised four nodes, three predicting the next the state of the Segmentation Network, and the fourth predicting the number of steps it will be in that state. This SRN also had a momentum of $\epsilon = 2.0$ and a momentum of 0.0. Training lasted for $10,000,000$ time steps.

After such training, Nolfi et al. made one of two changes to the robot's state. In one case, they moved it to a new position along the wall-following trajectory. In the other, they replaced the world with a slight variant. In both cases, the prediction error of the Upper SRN was very high immediately following the change, but after several cycles around the trajectory, it returned to a very low values. In a certain sense, then, the architecture has learned high-order representations of unique positions in the world.

This architecture has no mechanism for reporting current high-order state. However, in a separate test, Nolfi et al. added to the output layer two nodes trained to predict one of the two basis vectors corresponding to position close to either the left or right walls of the world and the zero vector corresponding to a position close to neither. The fully trained network properly predicted in which of these states it was in by activating the proper nodes. However, in this case, the experimenters applied a teacher to train the architecture. As we have noted, however, we wish to work toward an architecture that does not require a teacher, but nonetheless extracts meaningful symbols.

3.3.4 Problems

Nolfi and Tani's hierarchical architecture succeed in preliminary tests for time-dependent localization, and carries promise for improvements that move it toward more general goals. Currently, the architecture suffers two majors problems.

The primary problem in the hierarchical architecture is that the experiment produces reproducible results only when the Segmentation Network has three output nodes. Recall that we already noted that a both a human would likely and the RAQV did segment the robot's states into three symbols. However, if we are to achieve a world-independent solution, we must choose between three possibilities. In the first case, the architecture dynamically determines a reasonable number of symbols. In the second case, we fix the number of symbols, so that we must, before learning begins, give the architecture the number of symbols into which we want it to divide the world, and it must determine that number of symbols most meaningfully different from each other. Finally, in the third case, we predetermine an approximate number of symbols, but we leave it to dynamically determine the exact number.

The RAQV takes the first approach, to dynamically determine the number of symbols, and this seems more useful then the second, in which the number of symbols is fixed. However, we can also use architectures that take the second approach especially if in addition to a set of symbols and the ability to localize to them, we have a measure of the similarity between two symbols. Unfortunately, the Segmentation Network takes none of these approaches because it does not produce stable segmentation when the wrong number of symbols is chosen. In contrast, a human subject of whom it was demanded to describe four symbols, could invent a segmentation that makes sense as a set of abstract symbols that embody a world understanding. Thus, we fail to see the Segmentation Network as a viable architecture, or even part of one, for symbol localization.

The secondary problem is that we must choose criteria for determining when to switch phases, and the parameters to apply to the various components of the architecture. With so much uncertainty in design parameters, it is very challenging to develop and a general architecture. Possibly, we can use a genetic algorithm or similar approach to develop a general solution, but since the variety of worlds is so great and undetermined, and because architecture takes so long to test, this project is daunting and uncertain.

3.4 The Perks and Downs of Existing Architectures

We now review the main advantages and problems with the architectures we have discussed, and attempt to resolve the tensions between them.

Under restricted conditions the Segmentation Network of Nolfi et al. succeeds as a part of the of the hierarchy architecture, but has several important drawbacks. First, it demands 100,000 time steps for complete training, so the robot will be less quickly ready to make decisions. Second, it is based on a winner-takes-all system, so that the information it provides to components higher in the hierarchy is more limited. Third, the number of symbols is predetermined and fixed, even if that number is not suitable for the current world. Forth, and most import, the number of output nodes must be compatible with the specific world, in order for the network to produce results consistent across experiments and stable through time. The Segmentation Network is clearly not a good long-term strategy for our general goal.

A meaningful and reasonable number of nodes appears dynamically in the Resource Allocating Vector Quantizer. More, RAQV does not feature a separate training phase and does not ossify in time, so it responds to any change in the world at any time. However, the RAQV prohibits designers from specifying the approximate number symbols. Rather it forces them to specify their desired difference between symbols. While useful for some purposes, this situation opens the possibilities for the set of symbols expanding to an unmanageably large size, and also that of never surpassing a uselessly small size. Aware of this danger, we prefer a mechanism in which the approximate number of symbols is predetermined.

The Self-Organizing Map provides this feature because we merely specify the geometry and the distribution of model vectors and the frequency of each node's winning emerges un-supervised. However, the SOM is not ideally suited for his problem because it requires us to decide how to interpret the information it provides, and often fills completely, so that original number of symbols is the same as the number of nodes. If we wish we can is some cases assign nearby nodes to the same symbol, but doing that leaves us to determine a criterion for those cases. That criterion, however, need not be overwhelmingly complicated, as we can speculate reasonably that we can develop a solution.

From what we know, none of these architectures alone are capable of extracting high-order localization symbols. Very importantly, the RAQV and SOM are time-independent, so they have no information of previous state. The SRN carries a temporal memory, however it does not produce output of a meaningful symbol in a discrete space. We have argued throughout, as have Nolfi et al., that in order to find high-order symbols we must use combine several developmental architectures in a hierarchical arrangements. Such hierarchies generally demand more training time and system resources than single-component architectures, but if we consider humans as model example of developmental intelligence, we realize high-order reasoning takes long to develop. Indeed, we well know that children learn simplest elements of language first. First they learn syllables, then words, phrases, sentences, and much later to organize complicated thoughts into a long sequence of smaller ideas. Not until formal education do they learn to organize complicated concepts into abstract symbols. It is thus not remarkably disappointing that an architecture able to perform high-order symbol grounding is slow to learn. Hierarchy architecture, despite that drawback, carries promise in reaching a general solution to symbol localization.

4 Proposition of Self-Organizing Map for Improved Segmentation

We propose the a hierarchical architecture similar to Nolfi and Tani's except with a Self-Organizing Map [1] in place of the Segmentation Network. We provide the reasons for this choice along with a proposed experiment to test it.

4.1 Motivation

A Self-Organizing Map, developed by Teuvo Kohonen, is a developmental architecture for for the mapping of input vectors in continuous space of arbitrary dimension into a matrix element, or a node of the SOM. They train according to a scheme of unsupervised learning, so that the mapping of a given input is determined exclusively by past inputs. Kohonen demonstrated that the mapping, after proper training, is such that input vectors with similar abstract traits map to the same or nearby nodes. Thus, the SOM can place high-dimensional vectors in abstract categories. Kohonen assigns the term *feature space* to the positions of the matrix of nodes, since the SOM extracts features of the input vectors not apparent from non-abstract analysis.

Kohonen performed two experiments to demonstrate that the Self-Organizing Map distributes input into meaningful categories. In the first experiment he assigned to thirty-two symbols random vectors each of five elements having whole number values less than or equal to 6. He trained the SOM on these vectors and compared the SOM mapping to a minimum spanning tree of the same data. The number of edges in the path between two nodes of a minimum spanning tree is known to be smaller for more closely related vectors. The proximity of the nodes onto which the SOM mapped any two vectors corresponded closely to their proximity in the minimum spanning tree. Thus, Kohonen says, we should believe that the SOM exposes the categoric similarity of input vectors.

The second experiment, involving the representation of animals, is more convincing because it tests the SOMs on data representing real objects for which we already have a good abstract understanding. Here, Kohonen trained a SOM on vectors of normalized binary elements encoding fourteen basic traits of eighteen common animals falling into three general categories: mammalian herbivore, mammalian hunters, and birds. The input vectors representing the animals mapped to three clearly discernible regions in the SOM matrix, one for each of the three categories. The results also featured interesting properties, such as that the hunting birds fell closer to the mammalian hunters than did the non-hunting birds.

4.2 Self-Organizing Map for Very General Segmentation

The second example in the last section, if not the first, demonstrates that the SOM extracts abstract categories from complicated data. With this observation in mind, we find it reasonable to speculate that it could segment the hidden-layer activations of the Simple Recurrent Network in Nolfi and Tani's experiments so as to perform symbol localization. Because of the nature of the SOM, some nodes will be more populated than others. Some sparsely populated nodes may have limited meaning, but we hope that the highly populated nodes will represent important and meaningful localization symbols. It is unclear whether, in general, we will want to assign a separate symbol to each node, or whether we will want a technique to group nodes into the same symbol. This decision would affect how the SRN-SOM hierarchy is classified according to the distinctions we made in Section 3.3.4. If we give each node a unique symbol, then we have fixed the number of symbols and placed the SOM in the first category. If we group nodes into a single symbol depending on the emergent structure of the SOM, then the exact number of nodes is dynamically determined but the approximate number is predetermined.

We humans divide the world into a number of symbols that depends both on the approximate desired number of symbols and on the structure of the world, so that we fall into the third category. If we want robot architectures that are friendly to us, we should prefer a system that functions similarly. Thus, if we request from a robot in the the Two-Room World and a very small number of low-order symbols, we wold hope it to extract three symbols corresponding to straight walls, corners, and the corridor. If we request about ten high-order symbols, we would hope for symbols corresponding to about ten unique regions.

An architecture that answer such general questions is one that satisfies our long-term goal of a world-independent, time-dependent symbol self-localizer.

But this goal is too general for us to address in current experiments. Nevertheless, we can ask several experimental questions that would help us move toward it.

4.3 Self-Organizing Map for the Current Experiment

In our experiments, we ask three important questions of an architecture similar to Nolfi and Tani hierarchical architecture but with a Self-Organizing Map. Also, our experiments lack the Upper Simple Recurrent Network, but leave the possibility of feeding information about the winning node of the SOM to an SRN.

For our human curiosity, we make our first question whether the SOM will provide all the information of Nolfi et al.'s successful Segmentation Network. We will not attempt a systematic method to extract a small number of symbols from a SOM of a larger number of nodes. However, we visually compare the winning node to the absolute bearing and position of the robot.

Our second question is of what order will be the symbols corresponding to the winning nodes. Since the SOM input of the hidden-layer activations of the SRN encodes time-dependent information, it is reasonable to speculate that the SOM will immediately extract high-order symbols. However, the Segmentation Network did not find the higher order symbols. It will be important to know whether this is a failing of the Segmentation Network or of the information in its input. It is also important to know whether, if we built a hierarchical architecture involving a SOM, we would require a higher component to perform high-order symbol grounding. We thus ask, will a SOM node that indicates a indicates a straight wall or corner also indicate which one?

Finally we ask to what extent will the Euclidean distance between model vectors represent localization similarities between them. Euclidean distance between model vectors is frequently considered to test their conceptual similarity, and we speculate that if, in the future, we wished to have a method to group a cluster of SOM nodes into one symbol, Euclidean distance would form a part of the criterion.

Although we perform a relatively simple experiment and analyses, they answer questions these of crucial importance to the problem of building a hierarchical architecture more general than what Nolfi et al. have given us.

5 Experiment

Exactly as did Nolfi et al. and Kröse et al., we ran a wall-following direct control counter-clockwise around the two room world in a simulator. We used the Player / Stage simulator because of its availability, with the simulated USC Pioneer robot. However, for experimental control, we wished to simulate the Khepera robot, so we reduced the sixteen Pioneer sensors to eight virtual sensors, with ranges restricted to twice the length of the robot.

We applied an architecture in which a Simple Recurrent Network identical to Nolfi et al.'s was trained in the same manner during the first phase. Again, the network had input, hidden, and output layers of eight, three, and ten nodes, learning rate $\epsilon = 2.0$, and momentum 0.0. And again, we trained for $100,000$ time steps.

During the second phase, we trained a four-row three-column Self-Organizing Map using the hidden layer activations of the first level SRN as input to the SOM, much like Nolfi et al. used those activations as input to the segmentation network. We initialized the SOM with the last 300 hidden-layer activations and trained it with the next three-hundred. Including training, we collected records of the SOM winners of 10,000 time steps including the training steps.

Our architecture is illustrated in Figure 2.

6 Results

The SRN sum-squared error diminished with time, as shown in Figure 3, indicating that the network trained properly. The after about 5000 time steps, the error reached a steady value below 0.2. After about 2000 steps, we error begins to rise very slightly at a near-constant rate. We propose no explanation for this observation.

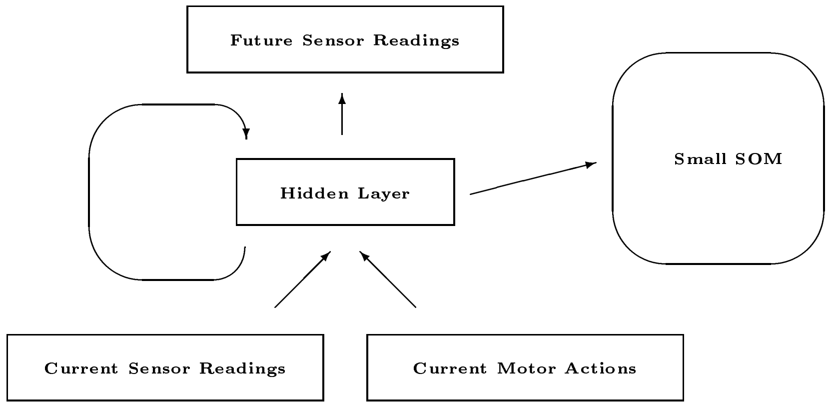

Figure 2: The architecture for our experiment. The current and future sensory information are encoded in eight nodes and the future motor action is encoded in two. The hidden layer has three nodes. The network has a learning rate of $\epsilon = 2.0$ and a momentum of 0.0. The SOM has four columns and three rows of nodes. It was initialized with 300 vectors.

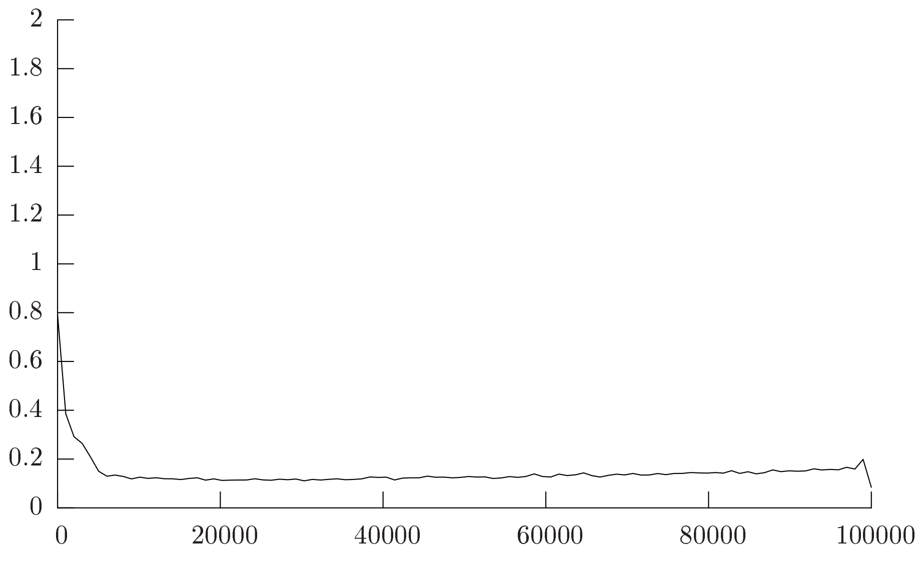

Figure 3: Sum squared error of the Simple Recurrent Network in on trial against time.

The behavior of our fully trained SOMs in two randomly selected experiments, and the Euclidean distances between model vectors of adjacent nodes, can be seen in Figures 4 and 5. A complete table of Euclidean distances for the same experiments can be found in Figures 6 and 7

Regarding the low-order localization distinction between corners and straight walls in the big room of the Two Room World of Nolfi and Tani's Segmentation Network. Each well-populated node of our SOMs clearly corresponds to corner-turning in the big room, wall-following there, or neither. Moreover, the first and second halves of each legs of these parts of the trajectory tend to map to separate nodes. For instance the SOM can tell us whether the robot, during straight wall-following, is approaching or leaving the nearest corner. This effect appears to be an instance of high-order localization. We see in Figure 8, that during a representative portion of a cycle in the trial represented in Figure 5, the differences in sensory state between times when nodes the selected node is either $(1, 2)$ and $(4, 1)$ is not significant. Both nodes correspond to straight wall-following, but the prior node corresponds to the first leg the trajectory along a wall, the latter node to second. The SOM clearly receives time-dependent information to make localization distinctions beyond that information available from current sensory-motor state. Thus the system performs some high-order localization.

A second example of high-order localization is clearer to see. Turning in the small room mapped to a distinct node from the turning in the big room. We did not observe straight wall-following in the small room. [1]ur wall-following direct control was such that the robot would constantly turn, for otherwise, it would be prone to come close enough to a wall that it might become stuck against. This reflects our use of a simulated USC Pioneer robot rather than a Khepera because of availability of the simulation tools. A Khepera is able to turn in place well, while a Pioneer is not.

Corner nodes and those adjacent to two corners were most populated; inner nodes generally were winners only a trivial number of times.

[1]O

Proximity on the SOM did not reflect similarity in location or action, nor did Euclidean distances between corresponding model vectors.

7 Discussion

The observation most important to our study is that the SRN-SOM hierarchical architecture abstracts symbols that provide meaningful information about the robot's location, but in a messy manner unfit to directly provide set of symbols that reliably partitions the world into meaningful regions. The SOM produces no guarantee that two states we humans consider insignificantly different will be mapped to the same node or that two we consider importantly distinct will be mapped to different nodes.

Further, neither adjacency on the SOM matrix nor Euclidean distance between model vectors provides a good indication of symbol similarity as we humans abstract the world. This is clear in Figure 4, in which nodes $(1, 3)$ and $(4, 3)$, which both represent straight wall-following, are separated by a distance of 0.43, and appear on opposite sides of the SOM: while $(1, 2)$ and $(1, 3)$, though they are only 0.23 apart in model space, are adjacent on the SOM matrix, and represent actions close spatially and temporally, they represent different symbols in three-symbol languages found by Nolfi et al.'s Segmentation Network and Lanåker et al.'s RAQV.

We have also shown our SRN-SOM to perform some high-order localization. It was surprising in Nolfi et al.'s results that the Segmentation Network, a representation of hidden-layer activations of an SRN, provided no time-dependent information. The localization of SRN-SOM is not full high-order localization because most symbols map to classes of regions, not unique regions. Thus, to perform high-order localization we must modify it, add to it, or replace it.

Nevertheless, the SRN-SOM does symbol localize, however noisy and low-order the results, and we can assign to each of the well-populated nodes natural-language phrases that describe them. In this sense, the architecture goes considerable length to make sense of the world, and gives us data substantially

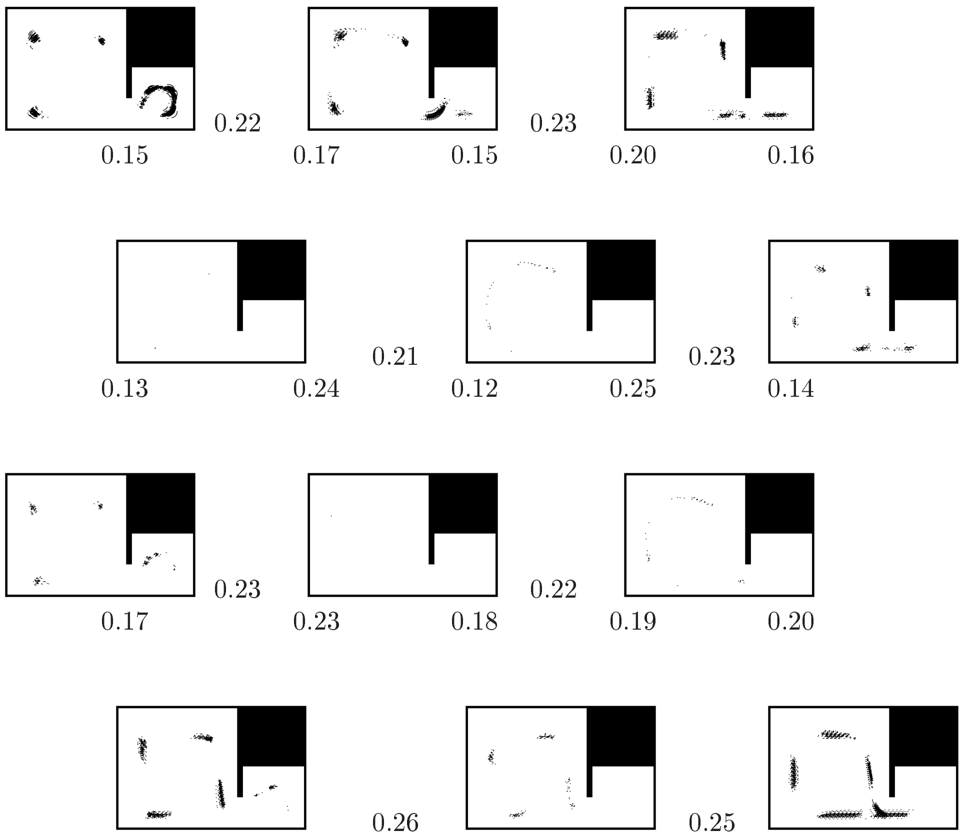

Figure 4: Self-Organizing Map segmentation of hidden-layer activations, and Euclidean distances of adjacent nodes. A complete table of Euclidean distances is given in Figure 6.

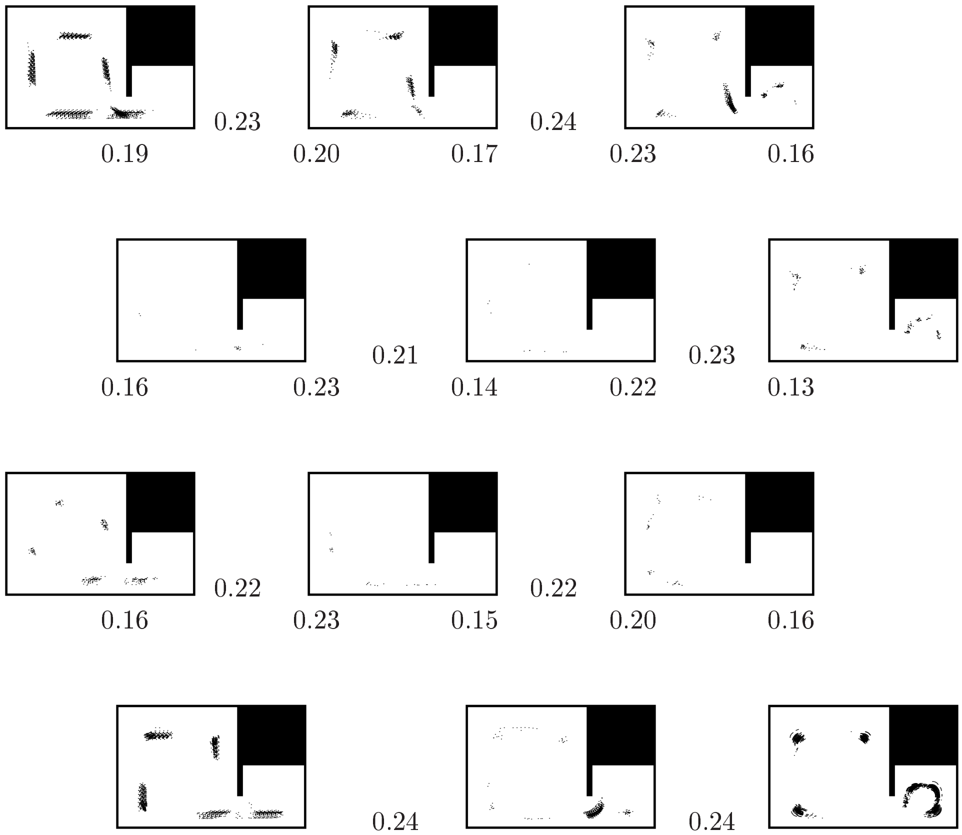

Figure 5: SOM data for another trial. All Euclidean distances given in Figure 7.

	(1, 1)	**(1, 2)**	**(1, 3)**	**(2, 1)**	**(2, 2)**	**(2, 3)**	**(3, 1)**	**(3, 2)**	**(3, 3)**	**(4, 1)**	**(4, 2)**	**(4, 3)**
(1, 1)		0.22	0.45	0.15	0.33	0.55	0.25	0.39	0.58	0.42	0.56	0.76
(1, 2)			0.23	0.17	0.15	0.34	0.28	0.26	0.40	0.41	0.43	0.59
(1, 3)				0.38	0.20	0.16	0.46	0.30	0.28	0.52	0.42	0.47
(2, 1)					0.21	0.44	0.13	0.24	0.44	0.28	0.41	0.62
(2, 2)						0.23	0.27	0.12	0.25	0.33	0.29	0.44
(2, 3)							0.48	0.27	0.14	0.49	0.31	0.31
(3, 1)								0.23	0.45	0.17	0.36	0.60
(3, 2)									0.22	0.23	0.18	0.38
(3, 3)										0.41	0.19	0.20
(4, 1)											0.26	0.51
(4, 2)												0.25
(4, 3)												

Figure 6: Euclidean distances between all pairs of nodes corresponding to the SOM given in Figure 4.

	(1, 1)	**(1, 2)**	**(1, 3)**	**(2, 1)**	**(2, 2)**	**(2, 3)**	**(3, 1)**	**(3, 2)**	**(3, 3)**	**(4, 1)**	**(4, 2)**	**(4, 3)**
(1, 1)		0.23	0.47	0.19	0.35	0.56	0.32	0.40	0.57	0.48	0.55	0.72
(1, 2)			0.24	0.20	0.17	0.34	0.35	0.29	0.38	0.47	0.43	0.53
(1, 3)				0.41	0.23	0.16	0.52	0.35	0.27	0.58	0.43	0.41
(2, 1)					0.21	0.45	0.16	0.23	0.42	0.31	0.37	0.56
(2, 2)						0.23	0.29	0.14	0.22	0.36	0.27	0.38
(2, 3)							0.51	0.30	0.13	0.53	0.33	0.25
(3, 1)								0.22	0.44	0.16	0.32	0.54
(3, 2)									0.22	0.23	0.15	0.34
(3, 3)										0.42	0.20	0.16
(4, 1)											0.24	0.48
(4, 2)												0.24
(4, 3)												

Figure 7: Euclidean distances for the SOM in Figure 5.

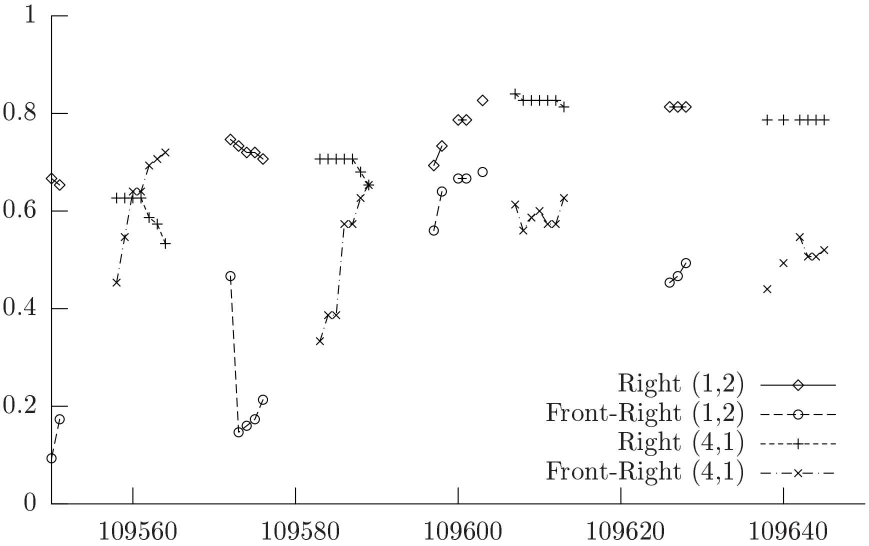

Figure 8: Normalized sensor values during selection of the $(1, 2)$ and $(4, 1)$ nodes for a representative part of a cycle. The second to the last segment of data corresponds to the selection of node $(1, 2)$, the last to node $(4, 1)$. Yet the sensory states at both times are nearly the same. Thus, the hierarchical architecture makes localization distinctions beyond those possible using only current sensory state.

more meaningful than the raw sensory-motor data. In the same way that the two-level SRN-SOM architecture transformed extremely noisy and chaotic data into a slightly noisy set of symbols, a higher-level addition to the architecture might transform that result into a clean set of symbols.

8 Conclusion

8.1 Analysis

A hierarchical architecture is better suited than a single-component architecture for the task of high-order symbol self-localization because different components perform different parts of the abstraction necessary to translate raw sensory-motor data into a language of clean and meaningful high-order symbols. A Self-Organizing Map partitions input in a time independent manner; a Simple Recurrent Network develops time-dependent abstractions without partitioning information into a discrete set of symbols. High-order localization depends on both time-dependence and on symbol segmentation, so no architecture can succeed with either component alone.

A SOM alone cannot high-order symbol-localize, but it may serve as a component of hierarchical system that can. The time-dependence of the results of our SRN-SOM architecture demonstrate this point. The system's segmentation is too noisy to constitute an acceptable localization language but clean enough to be further abstracted. Two possibilities for such further abstraction are to combine clusters of nodes into single symbols and to put more developmental components on top of the existing architecture.

Our results demonstrate that Euclidean distance and matrix proximity provide a poor indication of the abstract similarity between model vectors of SOM nodes. Thus, the time-independent state of the SOM does not provide adequate information to infer the symbolic relations between nodes. This want precludes applying a deterministic algorithm to decide how to group the many nodes of the SOM. Yet it leaves open the possibility for providing input to higher levels of a hierarchical architecture of developmental components. Developmental components are particularly suited for this task because they can use time recurrence to learn the relationship between nodes in each case of a fully trained SOM, rather than depending on its time-independent state and a predetermined computational algorithm.

We thus assert that an architecture that can high-order symbol-localize may feature our SRN-SOM architecture and place above it more components for greater abstraction toward high-order localization.

8.2 Future Work

Our architecture comprising Self-Organizing Map above the Simple Recurrent Network proved effective in organizing raw sensory-motor data, by nature extremely noisy, into a set of generally meaningful symbols, but not into a coherent, well-planned, and completely high-order set. Thus, we propose for the future the training of a second higher SRN to predict the next winning node of the SOM, similar to the manner in which Nolfi et al. trained the Upper SRN to predict the next state of the Segmentation Network. In that test, Nolfi et al. provided an extra node in both the input and output layers that indicated the length of time the Segmentation Network was in a particular state. If we too used a high-level SRN, we would experiment with this technique, but we would hope that we could achieve positive results simply by providing as input the winning node and allowing the time recurrence of the SRN to learn the time-dependent information. The second part of our proposed experiment, also features a higher-level SOM which receives as inputs the hidden-layer activations of the higher-level SRN. We would hope that that SOM would develop a more meaningful and high-order set of symbols than lower one. We believe that in light of our sound theoretical and experimental confidence in the hierarchical approach as a general solution to the problem, this test is of principle importance. Nevertheless, neither Nolfi et al. nor we have yet performed it.

However, the SRN-SOM may merit further testing before it is extended. To fully understand the architecture's efficacy, we must extensively test two features of its behavior: its consistency across trials and its similarity to our expectations. We provided our subjective analysis of a few randomly selected experimental trials, but no analysis objective or entailing a statistically significant number of runs. We thus require, for future investigation, systematic methods of testing the system's function. One approach for low-order symbols would be to divide the world into regions of local similarity, say the eight walls, to compute the mean and standard deviation of position of each location at which each symbol was reported, and to compare these figures with those from other experiments and from the regions into which a human might divide the world. This would might work reasonably well for rectangular regions, but probably would not for irregularly shaped ones. An alternative would be to divide the area close to the wall, along the robot's trajectory, into very small regions, record the number of times the SOM selected each node in each region, and to test for correlations across trials and against the human- selected partitioning. In one of these ways or a similar one, we could understand the consistency of the system's behavior.

A principle flaw in both Nolfi et al.'s and our system is that they provide current motor action as input to the low-level SRN. The first of two problems with this technique is that it depends on the availability of consistent motor information. Such would be unavailable if the robot's navigation were developmental, so that the motor response to sensory changed over the robot's lifetime. The second problem is that it allows the SRN to depend on the consistent motor action of the direct control to successfully learn the time-dependent nature of its movement through the world, so that the SRN may not be performing true localization as we generally understand it, the use of past and current sensory information to determine current location. Rather the SRN also receives information from the direct-control algorithm's interpretation of sensory input. The system would be more flexible if this non-developmental step could be eliminated. The necessity of the motor-action input is an issue that can be approached experimentally.

The final avenue of future work we propose is to test the same architecture in different worlds and undergoing different behaviors. We and Nolfi et al. both produced some positive results, but under very controlled circumstances. The Two Room World was optimized for the experiment, and the robots followed the same cyclic trajectory for an extended period. Thus, more complete research would alter these conditions, perhaps to some less friendly to the robot. Such work would help us to move toward our goal of a world-independent system for symbol-localization. Stout and Tan have already begun investigate the behavior of an architecture similar to ours but in other, more general worlds. Unfortunately, their results are not yet available. As researchers both provide answers to pending experimental questions and provide fresh theoretical insight to the problem of symbol self-localization, we will grow closer to our goal of developing a high-order world-independent solution to that grand problem.

Special Thanks

I thank Lisa Meeden for her sound and experienced advice and for her careful and thorough teaching. I also thank Andrew Stout and Yee Lin Tan for their effective corroboration in ideas and exchange of technical assistance without stepping over boundaries of creative domain.

References

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

[2] B. Krose and M. Eecen. A self-organizing representation of sensor space for mobile robot navigation, 1994.

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

[4] 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.