Learning Navigational Behaviors using a Predictive Sparse Distributed Memory[*]

Rajesh P.N. Rao and Olac Fuentes
Department of Computer Science
University of Rochester
Rochester, NY 14627
{rao,fuentes}@cs.rochester.edu

Abstract

We describe a general framework for the acquisition of perception-based navigational behaviors in autonomous mobile robots. A self-organizing sparse distributed memory equivalent to a three-layered neural network is used to learn the desired transfer function mapping sensory input into motor commands. The memory is initially trained by teleoperating the robot on a small number of paths within a given domain of interest. During training, the vectors in the sensory space as well as the motor space are continually adapted using a form of competitive learning to yield basis vectors aimed at efficiently spanning the sensorimotor space. After training, the robot navigates from arbitrary locations to a desired goal location using motor output vectors computed by a saliency-based weighted averaging scheme. The pervasive problem of perceptual aliasing in non-Markov environments is handled by allowing both current as well as the set of immediately preceding perceptual inputs to predict the motor output vector for the current time instant. Simulation results obtained for a mobile robot, equipped with simple photoreceptors and infrared receivers, navigating within an enclosed obstacle-ridden arena indicate that the method performs successfully in a variety of navigational tasks, some of which exhibit substantial perceptual aliasing.

1 Introduction

Behaviors that realize goal-directed, collision-free navigation in unstructured environments form an extremely important component of the behavioral repertoire of any autonomous mobile robot. Traditional sensorimotor controllers relied on fixed robot behaviors that were prewired by the control designer. Such an approach suffered from at least two problems: (a) the increasing complexity of the design process when scaling up from toy-problems to real environments and (b) the inherent inflexibility of prewired behaviors due to their inability to adapt to circumstances unforeseen at design time. The first problem is addressed by Brooks [4], who proposes a *hierarchical behavior-based decomposition* of control architectures. Such behavior-based robot architectures have been shown to accomplish a wide variety of tasks in an efficient manner [7, 19]. The second problem has been addressed by endowing robots with the ability to autonomously learn behaviors either from experimentation and dynamic interaction with their environment or via teleoperation. A number of learning algorithms have been used for this purpose including neural networks [29, 30, 37], evolutionary algorithms/genetic programming [3, 6, 18], reinforcement learning [2, 13, 20], hill-climbing routines [10, 27], and self-organizing maps [16, 24, 35].

In the context of robot navigation, a popular approach has been the construction and use of global maps of the environment [9]. Such an approach, while being intuitively appealing, faces the difficult problem of robot localization *i.e.* establishing reliable correspondences between noisy sensor readings and geometrical map information in order to estimate current map position. In addition, the global map inherits many of the undesirable properties of centralized representations that Brooks [4] identifies in traditional robot controllers. An alternative behavior-based approach to navigation is proposed by Mataric [22] (see also [8, 17]). This method avoids the problems involved in creating and maintaining global representations by utilizing only local information as provided by landmarks along a navigational path. A potential problem with such landmark-based approaches is that incorrect landmark matches or changes in the landmarks themselves might result in catastrophic failure of the navigational system.

In this paper, we propose a robust behavior-based approach to goal-directed mobile robot navigation based on a predictive sparse distributed memory. A “teaching-by-showing” paradigm is adopted to initially train the memory for achieving a prespecified navigational behavior; such an approach drastically cuts down on the number of training trials needed

[*]To appear in the Proc. of *From Animals to Animats: The Fourth International Conference on Simulation of Adaptive Behavior*, MIT Press, 1996.

as compared to learning by trial and error or by random exploration of the environment. In our case, the robot is teleoperated (via a remote joystick) on a small number of training paths within the given environment. During training, the vectors in the perception space as well as the motor space are adapted using a form of competitive learning that aims to construct basis vectors that efficiently span the sensorimotor space encountered by the robot. After training, the robot navigates from arbitrary locations to a desired goal location based on motor output vectors computed by indexing into sensorimotor memory with present as well as past perceptions, and using a saliency-based weighted averaging scheme for interpolating output vectors. By using past sensory inputs to provide the necessary context for disambiguating potentially similar perceptions, we alleviate the well-known problem of perceptual aliasing [5, 38] in non-Markov environments. We provide simulation results for a mobile robot, equipped with simple photoreceptors and infrared receivers, navigating within an enclosed obstacle-ridden arena. The method is shown to perform successfully in a variety of navigational tasks, some of which exhibit substantial perceptual aliasing.

2 Sparse Distributed Memory

One possible method for memory-based navigation is to use an associative memory in the form of a look-up table that associates a large set of perception vectors from different locations with corresponding set of actions required to navigate to a desired location [25]. However, such an approach suffers from at least three drawbacks: (a) the address space formed by the perception vectors is usually quite large and therefore, storing fixed reference vectors for every possible scenario becomes infeasible; (b) the training time increases drastically as the look-up table size increases, and (c) simple look-up table strategies relying on nearest-neighbor methods usually fail to generate the appropriate responses to novel situations. In this section, we address problems (a) and (b) by using only a *sparse* subset of the perceptual address space. This naturally leads to a memory known as sparse distributed memory (SDM) that was originally proposed by Kanerva [14]. We address the second problem of generalization in novel scenarios in the next section where we propose a modified form of SDM that uses competitive learning to adapt its sensorimotor space and radial interpolation to compute motor output vectors.

2.1 Kanerva's Model

Sparse distributed memory (SDM) (Figure 1) was first proposed by Kanerva [14] as a model of human long-term memory. It is based on the crucial observation that if concepts or objects of interest are represented by high-dimensional vectors, they can benefit from the very favorable matching properties caused by the inherent tendency toward orthogonality in high-dimensional spaces.

Kanerva's SDM can be regarded as a generalized random-access memory wherein the memory addresses and data words come from high-dimensional vector spaces. As in a conventional random-access memory, there exists an array of storage locations, each identified by a number (the address of the location) with associated data being stored in these locations as binary words. However, unlike conventional random-access memories which are usually concerned with addresses only about 20 bits long (memory size = 2^{20} locations) and data words only about 32 bits long, SDM is designed to work with address and data vectors with much larger dimensions. Due to the astronomical size of the vector space spanned by the address vectors, it is physically impossible to build a memory containing every possible location of this space. However, it is also unnecessary since only a subset of the locations will ever be used in any application. This provides the primary motivation for Kanerva's model: *only a sparse subset of the address space is used for identifying data locations and input addresses are not required to match stored addresses exactly but to only lie within a specified distance of an address to activate that address.*

The basic operation of SDM as proposed by Kanerva can be summarized as follows:

1. **Initialization**: The physical locations in SDM correspond to the rows of an $m \times k$ contents matrix **C** (initially filled with zeroes) in which data vectors $\in \{-1, 1\}^k$ are to be stored (see Figure 1 (a)). Pick m unique addresses (p-element binary vectors $\mathbf{r}_i$) at random for each of these locations (these addresses are represented by the matrix **A** in Figure 1 (a)).

2. **Data Storage**: Given an p-element binary address vector $\mathbf{r}$ and a k-element data vector $\mathbf{d}$ for storage, select all storage locations whose addresses lie within a Hamming distance of D from $\mathbf{a}$ (these activated locations are given by the *select* vector $\mathbf{s}$ in Figure 1 (a)). *Add* the data vector $\mathbf{d}$ to the previous contents of each of the selected row vectors of **C**. Note that this is different from a conventional memory where addresses need to exactly match and previous contents are overwritten with new data.

3. **Data Retrieval**: Given an p-element binary address vector $\mathbf{r}$, select all storage locations whose addresses lie within a Hamming distance of D from $\mathbf{r}$ (these locations are again given by the vector $\mathbf{s}$). Add the values of these selected locations in parallel (*i.e.* vector addition) to yield a sum vector **S** containing the k sums. Threshold these k sums at 0 to obtain the data vector $\mathbf{d}'$ *i.e.* $d_i = 1$ if $s_i > 0$ and $d_i = -1$ otherwise.

Note that the addition step in (2) above is essentially a *Hebbian learning rule*. The statistically reconstructed data vector $\mathbf{d}'$ should be the same as the original data vector provided the *capacity* of the SDM [15] has not been exceeded. The intuitive reason for this is as follows: When storing a data vector $\mathbf{d}$ using an p-dimensional address vector $\mathbf{r}$, each of the selected locations receives one copy of the data. During retrieval with an address close to $\mathbf{r}$, say $\mathbf{r}'$, *most* of the locations that were

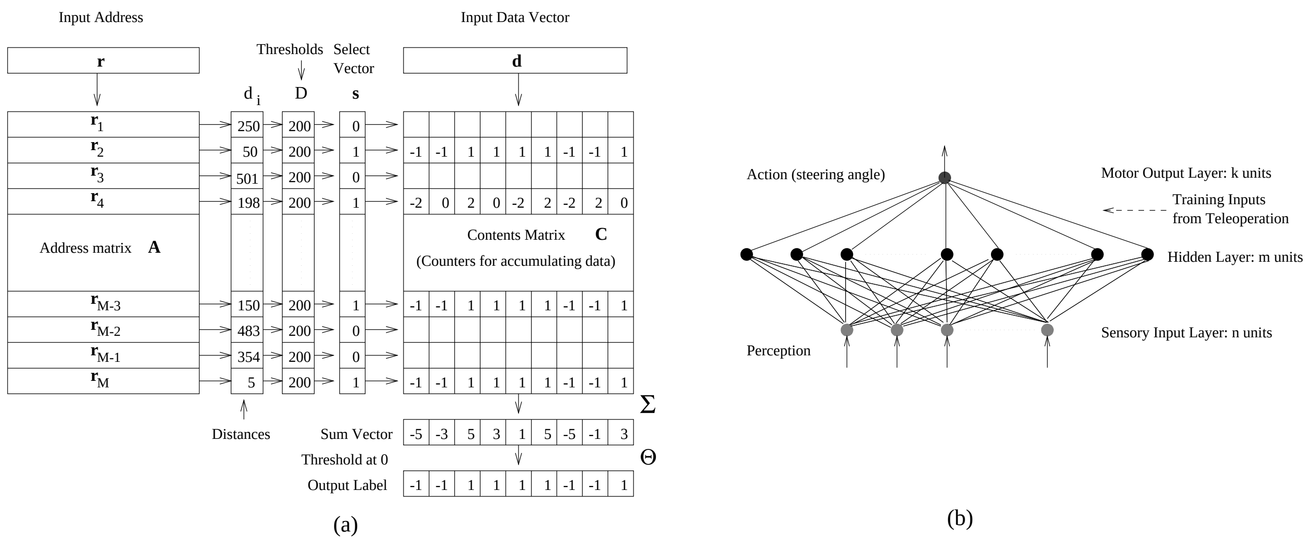


Figure 1: **Kanerva's Sparse Distributed Memory (SDM).** (a) shows a schematic depiction of the model as proposed by Kanerva for storage of binary (and bipolar) vectors. (b) shows a realization of the memory in the form of a three-layered fully connected network. The labels describe how the network can be used for perception-based navigation after suitable modifications (see Section 3).

selected with $\mathbf{r}$ are also selected with $\mathbf{r}'$. Thus, the sum vector contains most of the copies of $\mathbf{d}$, plus copies of other different words; however, due to the orthogonality of the address space for large p, these extraneous copies are much fewer than the number of copies of $\mathbf{d}$. This biases the sum vector in the direction of $\mathbf{d}$ and hence, $\mathbf{d}$ is output with high probability. A more rigorous argument based on signal-to-noise ratio analysis can be found in [15].

2.2 Limitations of Kanerva's Model

The model of sparse distributed memory as originally proposed by Kanerva has several weaknesses that prevent its direct use in memory-based navigation:

- Both the address and data vectors are required to be binary in the standard model of the SDM. Since most natural environments yield multivalued input patterns, we must either modify the indexing mechanisms of the model or recode all inputs into binary form. We chose the former option since the latter sacrifices the interpolation properties of the memory.

- The standard model of the SDM assumes a *uniform distribution* for the input address vectors whereas in most natural circumstances, the input address vectors tend to be clustered in many correlated groups distributed over a large portion of the multidimensional address space. Therefore, if addresses are picked randomly as suggested by Kanerva, a large number of locations will never be activated while a number of locations will be selected so often that their contents will resemble noise. We remedy this situation by allowing the input address space to *self-organize* according to the sensorimotor input distribution.

- The standard SDM uses a single fixed threshold D for activating address locations. While this simplifies the analysis of the memory considerably, it also results in poor performance since the actual values of the distances between an input address vector and the basis address vectors in the SDM are lost during quantization to 0 or 1. Our model uses *radial interpolation* functions that weight corresponding data vectors according to the address vector's closeness to the input vector.

3 A Self-Organizing SDM for Perception-Based Navigation

In the following, we describe the operation of a modified form of SDM that is suitable for memory-based navigation. The memory can be realized as a three-layer fully-connected feedforward network as shown in Figure 1 (b). Assume the memory contains m storage locations. The first layer consists of n units representing the input perception vector $\mathbf{p}$. The hidden layer consists of m units while the output layer is represented by k units. Let $\mathbf{w}_i$ ($1 \leq i \leq m$) represent the vector of weights between hidden unit i and the input layer, and let $\mathbf{m}_i$ represent the vector of weights between hidden unit i and the output layer. The memory accepts multivalued perception vectors $\mathbf{p}$ from an arbitrary distribution and stores an associated multivalued motor vector $\mathbf{m}$ in a distributed manner in the data space.

3.1 Initialization

Pick m unique addresses (n-dimensional vectors $\mathbf{p}_i$) at random for each of the locations. This corresponds to randomly initializing the input-hidden weight vectors $\mathbf{w}_i$ ($1 \leq i \leq m$).

3.2 Competitive Learning of Sensorimotor Basis Functions

Given an input perception vector $\mathbf{p}$ and an associated motor vector $\mathbf{m}$ during the training phase, we self-organize the address/data space using a *soft competitive learning rule* [26, 40]:

1. Calculate the Euclidean distances $d_j = ||\mathbf{w}_j^t - \mathbf{p}||$ between $\mathbf{w}_j^t$ and the input perception vector $\mathbf{p}$.

2. Adapt all weight vectors (sensory address space vectors) according to:

$$\mathbf{w}_j^{t+1} \leftarrow \mathbf{w}_j^t + g_j(t) P_t(d_j)(\mathbf{p} - \mathbf{w}_j^t) \qquad (1)$$

   where P_t is defined as:

$$P_t(d_j) = \frac{e^{-d_j^2/\lambda_j(t)}}{\sum_{k=1}^{m} e^{-d_k^2/\lambda_k(t)}} \qquad (2)$$

   and g_j is given by $g_j(t) = 1/n_j(t)$ where the counter:

$$n_j(t+1) = n_j(t) + P_t(d_j) \qquad (3)$$

   $P_t(d_j)$ can be interpreted as the probability of the prototype vector $\mathbf{w}_j$ winning the current competition for perception $\mathbf{p}$. Note that the probability vector $\mathbf{P}$ obtained by vectorizing the $P_t(d_j)$ for $1 \leq j \leq m$ is the equivalent of the *select* vector $\mathbf{s}$ in Kanerva's model (Figure 1 (a)).

   The "temperature" parameter $\lambda_j(t)$ is gradually decreased to a small value in a manner reminiscent of simulated annealing. This causes the learning algorithm to evolve from an initially soft form of competition with a large number of winners to the case where only a *sparse* number of winners exist for any given input vector. The soft competition in the beginning tunes the initially random prototype vectors towards the input sensory space (thereby preventing the occurrence of "dead" units which never get updated) while the later existence of sparse number of winners helps in fine tuning the prototype vectors to form a set of distributed *basis vectors* spanning the sensory input space.

3. Given a training motor input $\mathbf{m}$, adapt the prototype vectors stored in the motor space according to equation 1 using the *same* values for d_j as in 1 above (*i.e.* distance between the input perception vector and the sensory basis vectors $\mathbf{w}_j^t$):

$$\mathbf{m}_j^{t+1} \leftarrow \mathbf{m}_j^t + \beta_j(t) P_t(d_j)(\mathbf{m} - \mathbf{m}_j^t) \qquad (4)$$

   where $\beta_j(t)$ ($0 < \beta_j(t) < 1$) is a gain function. Note that $\beta_j(t)$ *does not necessarily* have to decrease with time (this reinforcement strength could be made to depend on other factors such as importance to the animate system as evaluated by other modalities or other internal value mechanisms).

3.3 Computing Motor Output Vectors

Following training, the memory *interpolates* between the stored motor basis vectors to produce a motor output vector $\mathbf{o}$ for a given perception vector $\mathbf{p}$, as follows:

1. Calculate the Euclidean distances d_j between $\mathbf{w}_j$ and the input perception vector $\mathbf{p}$.

2. Let m_{ji} ($1 \leq i \leq k$) denote the weight from hidden unit j to output unit i. Then, the ith component of the reconstructed output vector $\mathbf{o}$ (in other words, the output of the output unit i) is given by:

$$o_i = \sum_{j=1}^{m} P_t(d_j) m_{ji} \qquad (5)$$

The saliency-based weighted averaging above is a form of normalized radial interpolation that is similar to the output operation of *radial basis function* (RBF) networks [28]: the closer the current perception is to a given sensory basis vector, the more "salient" that memory location and the more the weight assigned to the motor vector associated with that basis vector. The above scheme is inspired by recent neurophysiological evidence [23] that the *superior colliculus*, a multilayered neuronal structure in the brain stem that is known to play a crucial role in the generation of saccadic eye movements, in fact employs a population averaging scheme similar to the one above to compute saccadic motor vectors.[1]

4 Predictive Sensorimotor Memory

The self-organizing SDM described in the preceding section has been shown to be useful for perceptual homing by an autonomous mobile robot [35]. However, it is not hard to see that the above method is limited to only those navigational behaviors that can be modeled by *Markov* processes: the current motor output is dependent solely on the current perception and past inputs are treated as irrelevant for determining current action. In general environments, however, a Markovian assumption is often inappropriate, and any method that relies on such an assumption suffers from the problem of *perceptual aliasing* [5, 38].

4.1 Perceptual Aliasing

Perceptual aliasing, a term coined in [38], refers to the situation wherein two or more identical perceptual inputs require different responses from an autonomous system. A number of factors such as limited sensing capability, noisy sensory inputs, and restricted resolution in addition to inherent local ambiguity in typical environments contribute towards exacerbating perceptual aliasing. The effects of aliasing can be reduced to some extent by incorporating additional sensory information that suffice to disambiguate between any two given

[1]We refer the interested reader to an earlier paper [32] for a closely related method for visuomotor learning of saccadic eye movements for a robot head.

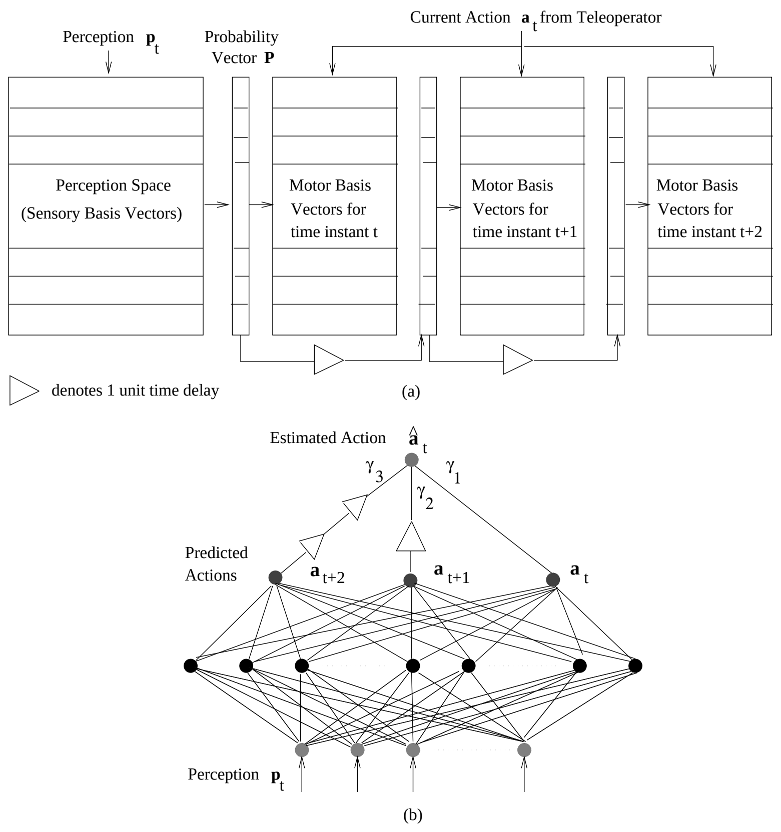

Current Action $\mathbf{a}_t$ from Teleoperator
Perception $\mathbf{p}_t$
Probability Vector $\mathbf{P}$
Perception Space (Sensory Basis Vectors)
Motor Basis Vectors for time instant t
Motor Basis Vectors for time instant t+1
Motor Basis Vectors for time instant t+2
denotes 1 unit time delay
(a)
Estimated Action $\hat{\mathbf{a}}_t$
γ_3
γ_2
γ_1
Predicted Actions
$\mathbf{a}_{t+2}$
$\mathbf{a}_{t+1}$
$\mathbf{a}_t$
Perception $\mathbf{p}_t$
(b)

Figure 2: **Predictive Sparse Distributed Memory.** (a) shows the procedure of training the predictive memory. The current motor input is fed to each of the motor memories which are indexed by the current perception (left), the previous one (middle), and the perception before the previous one (right). This allows the memory to generate predictions of motor output for the next two time steps in addition to the current one during the navigation phase. (b) shows the memory (in its network form) during the navigation phase. The current estimate of motor output $\hat{\mathbf{a}}_t$ is computed by averaging over the motor action vectors for the current time instant t predicted by current and past perceptual inputs.

situations [38, 39]. However, these methods still rely only on current percepts and thus, are unable to overcome aliasing in non-Markov environments. An alternative approach that is tailored toward modeling non-Markov sensorimotor processes is to base the current output on both the current as well as the sequence of past outputs (within a certain time window). In other words, the current percept not only predicts the action for the current time instant t but also actions for future time instant $t, t+1, \ldots, t+k$, where k is a parameter whose value can be either predetermined or adapted on-line to counteract the effects of aliasing. The motor action for the current time instant t is then determined by a weighted average of the actions recommended by the current as well as past perceptions upto time $t-k$. This solution shares some similarities with Kanerva's k-fold memory for storage and retrieval of sequences [14]; however, all the weaknesses inherent in Kanerva's original model (Section 2.2) still apply to the k-fold memories, thus preventing their direct application in countering the aliasing problem.

4.2 Using Past Perceptions for Motor Prediction

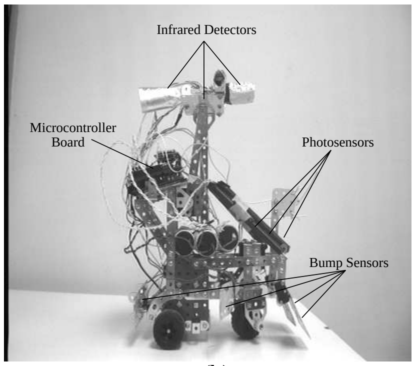
The informal solution for perceptual aliasing sketched in the previous section can be formalized as follows. Let $\mathbf{p}_t$ be the current perception vector and let $\mathbf{m}$ be the current motor action vector. Figure 2 shows the predictive memory. Note that the motor space now contains additional sets of connections between the hidden and output layer, each such set determining the action to be executed at a given time in the future. A further set of connections (see (b)), some of which involve time delays of different durations, appropriately combine the outputs of this intermediate layer to yield an estimate of current motor output. The operation of the predictive memory is as follows:

- **Training**: During training (Figure 2 (a)), the perception space is self-organized as given by Equation 1. The motor space is also self-organized as in Equation 4 but using the *current* motor vector as the training signal for *each* set of hidden to motor output unit connections, with the constraint that the probability weight vector $\mathbf{P}$ (see Section 3.2) for the set of connections determining the action for time $t+i$ is the one that was obtained by indexing into the sensory address space using the perception at time $t-i$. Such a training paradigm ensures that after training, the perception at time t generates predicted actions for time steps $t, t+1, \ldots, t+k$.

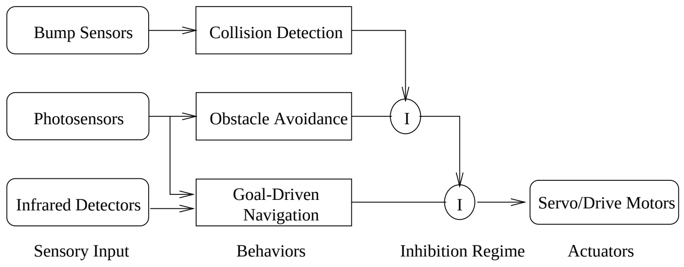
Destination Locations

		One	Many
Source Locations	One	Simple Interlocation Navigation	Example: Foraging
	Many	Example: Homing	General-Purpose Navigation

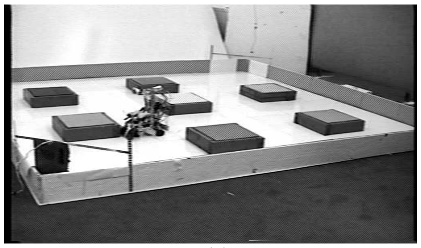
(a)



(b)



(d)



(c)

Figure 3: **Experimental Methodology**. (a) A hierarchical decomposition of the general navigation problem. Preliminary experimental results indicate that the robot is capable of acquiring each of the four cases of navigational behaviors shown in the diagram. (b) The mobile robot that inspired the simulations in the experiments. (c) The robot in its environment. (c) Robot control architecture: the collision detection and obstacle avoidance routines were autonomously learned on-line by the robot as described in [10]. "I" denotes inhibition of the current behavior by a lower-level behavior.

- **Navigation**: During the autonomous navigation phase (Figure 2 (b)), the current perception yields a set of actions $\mathbf{a}_t, \mathbf{a}_{t+1}, \ldots, \mathbf{a}_{t+k}$. Thus, at any given time instant t $(t \geq k)$, there exist k estimates $\mathbf{a}_t^1, \mathbf{a}_t^2, \ldots, \mathbf{a}_t^k$ of what the current action should be based on current and past sensory inputs. We obtain the combined estimate of current motor output using the following weighted averaging method:

$$\hat{\mathbf{a}}_t = \sum_{i=1}^{k} \gamma_i \mathbf{a}_t^i \qquad (6)$$

where the γ_i determine the fidelity of estimate i and can be experimentally determined or autonomously learned.

5 Experimental Results

The methods proposed in the previous sections were tested by simulating the behavior of an actual robot (Figure 3 (b)) in an enclosed obstacle-filled environment (Figure 3 (c)). This robot was previously used for on-line learning of a hierarchical set of behaviors [10] but the slow processing speed of the on-board microcontroller unfortunately limited its use in the present endeavor; we therefore evaluated the feasibility of the algorithms presented in this paper by using a simulation of the robot instead. The simulated robot was equipped with the same three classes of sensors as did its real world counterpart:

- **Bump Sensors**: Realized using digital microswitches, these sensors indicate whether the robot was physically touching an obstacle. Five of these sensors, placed at different locations around the robot, were used for the *obstacle detection* behavior.

- **Photosensors**: Six horizontally-positioned photoreceptors (implemented via shielded photoresistors) were employed for measuring the amount of light from a light source located near the arena. Six tilted photoreceptors were used for measuring light intensity value due to the color of the floor and for detecting surrounding obstacles. The outputs from these sensors were used for the *obstacle avoidance* behavior as well as for learning the perception-based navigational behaviors;

- **Infrared detectors**: These sensors, when used in conjunction with infrared detection software, indicate the strength of the modulated infrared light in a small spread along the line of sight of the sensor. Output from four of these sensors were used for learning the navigational behaviors.

The above sensory repertoire was supplemented by two effectors consisting of a rear drive motor and a servo motor at the front for steering the robot. For the simulations, robot perceptions were computed using the simplifying assumption that light/infrared intensity at a particular orientation θ and at a distance r from the source is proportional to the solid angle subtended by the source (*i.e.* $cos(\theta)/r^2$), assuming unit area for the robot receptors.

In the first set of experiments, the simple non-predictive self-organizing SDM from Section 3 was used for training the robot to home to a particular location from an arbitrary

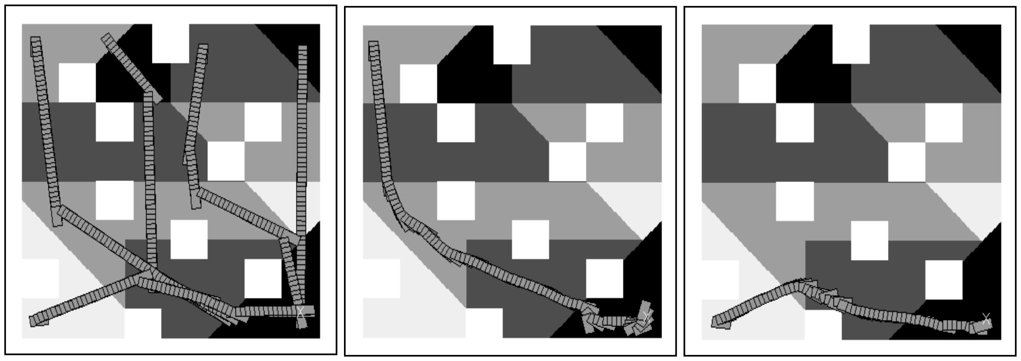

(a) (b) (c)

Figure 4: **Homing using the non-predictive Sparse Distributed Memory** (a) The paths on which the network was trained by teleoperating the robot. Floor color is depicted by shades of grey and obstacles are colored white. The home position is marked by an 'X'. (b) and (c) depict the paths chosen by the robot when placed at two different positions within the arena. Note the slight deviations from the training paths caused by mild perceptual aliasing; the obstacle avoidance behavior is automatically invoked when the deviations cause encounters with the wall or the square obstacles.

number of other locations in the robot arena (this corresponds to the many-one navigation box in the classification of Figure 3 (a)). Figure 4 shows some typical examples where the robot successfully navigates to the home position. Note that the existence of mild perceptual aliasing causes it to deviate on some occasions, thereby necessitating the use of the lower level obstacle avoidance behaviors.

In the second set of experiments (corresponding to the many-many box in Figure 3 (a)), the robot was trained on a number of paths that *intersected* each other at different locations (Figure 5 (a)). Thus, at these locations, local perceptions alone do not suffice to determine the steering direction, and the non-predictive memory usually fails to find the correct direction to continue in order to reach the pertinent goal destination. On the other hand, as shown in Figure 5 (b), (c) and (d), the predictive SDM allows the robot to use the context of past perceptions to determine its current action, thereby guiding it to its appropriate destination in spite of the presence of varying degrees of perceptual aliasing.

6 Summary and Conclusions

We have shown that a predictive sparse distributed memory provides an efficient platform for learning adaptive navigational behaviors. The proposed method enjoys severable favorable properties:

- **Sparse Memory**: In contrast to nearest-neighbor look-up table techniques, the present method employs only a sparse number of memory locations and avoids the "curse of dimensionality" problem by intelligently sampling the high-dimensional sensorimotor space using competitive learning (see below).

- **Competitive Learning of Sensorimotor Basis Functions**: The contents of the sparse memory are self-organized using a form of competitive learning that can be related to maximum likelihood estimation [26]. The learning rule allows the memory to autonomously form its own set of basis functions for describing the current sensorimotor space.

- **Distributed Storage**: Inputs to the memory are distributed across a number of locations, thereby inheriting the well-known advantages of a distributed representation [11] such as generalization to previously unknown inputs and resistance to faults in memory and internal noise.

- **Motor Prediction based on Past Perceptual History**: The problem of *perceptual aliasing* is alleviated by employing past perceptions to predict and influence current motor output. This extends the application of the method to non-Markov sensorimotor environments and distinguishes our approach from previous neural network approaches based on training procedures such as back-propagation [29].

- **Biological Plausibility**: The structure of the memory bears some striking similarities to the organization of the mammalian cerebellum (and therefore to the cerebellar model of Marr [21] and the CMAC of Albus [1]).[2] It is therefore plausible that biological structures and learning processes similar in spirit to those proposed herein may underlie goal-directed perception-based navigation in animals.

Sparse distributed memories have previously been used for a wide variety of tasks such as object recognition [34], face recognition [33], speech recognition [31], speech synthesis [12], and weather prediction [36]. The present work shows that with suitable modifications, such memories can be used for learning useful navigational behaviors in mobile robots as well. Ongoing work involves implementing the learning method on a recently acquired wheelchair robot and designing learning procedures for on-line adaptation of some of the free parameters in the current method such as the length of the perceptual history window and the fidelity γ_i for combining

[2]See [15, 35] for more details.

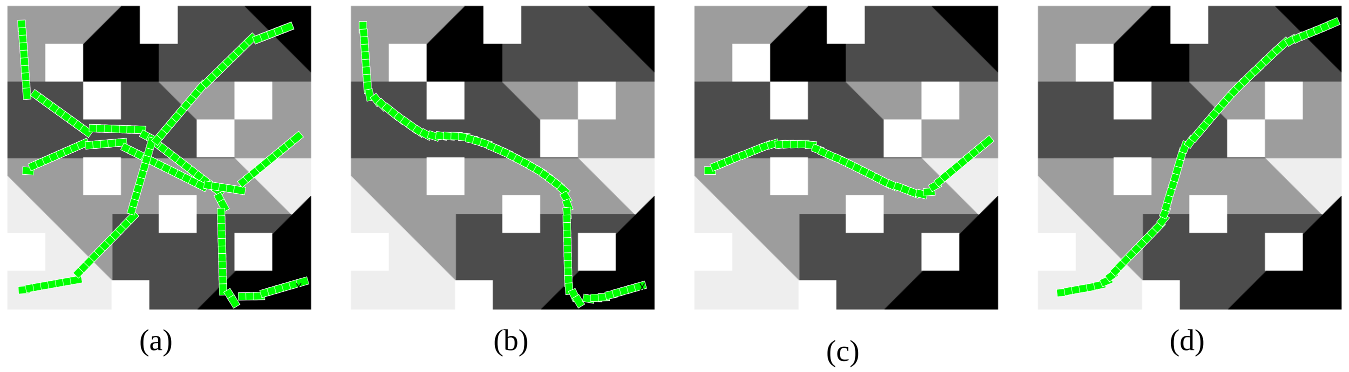


Figure 5: **Navigation using the Predictive Sparse Distributed Memory** (a) The paths on which the network was trained by teleoperating the robot. Note the intersection of the training paths at various points which gives rise to perceptual aliasing. (b), (c) and (d) show that the predictive memory is able to circumvent aliasing effects and follow the path to its goal destination by using past sensory information as a contextual aid to disambiguate aliased perceptions.

predicted actions. A simultaneous effort involves exploring the use of predictive recurrent networks in conjunction with learning rules other than competitive learning for generating useful behaviors in autonomous robots.

Acknowledgments

We would like to thank Dana Ballard for several interesting discussions, Chris Brown for his encouragement and support for building the mobile robot, and Mike Van Wie for his help in the construction of the robot.

References

[1] James S. Albus. A theory of cerebellar functions. *Math. Biosci.*, 10:25–61, 1971.

[2] M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, and K. Hosoda. Coordination of multiple behaviors acquired by a vision-based reinforcement learning. In *Proceedings of the 1994 IEEE/RSJ International Conference on Intelligent Robots an Systems*, pages 917–924, Munich, Germany, 1994.

[3] R.D. Beer and J.C. Gallagher. Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, 1:91–122, 1992.

[4] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–22, April 1986.

[5] Lonnie Chrisman. Reinforcement learning with perceptual aliasing. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, 1992.

[6] Dave Cliff, Philip Husbands, and Inman Harvey. Evolving visually guided robots. In J. A. Meyer, H. Roitblat, and S. Wilson, editors, *From Animals to Animats 2: Proceedings of the Second International Conference on the Simulation of Adaptive Behavior*, pages 374–383. Cambridge, MA: MIT Press, December 1992.

[7] Jonathan H. Connell. *Minimalist mobile robotics : A colony-style architecture for an artificial creature*. Academic Press, Boston, MA, 1990.

[8] P.G.R. de Bourcier. Animate navigation using visual landmarks. Cognitive Science Research Papers 277, University of Sussex at Brighton, May 1993.

[9] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, 3:249–265, 1987.

[10] Olac Fuentes, Rajesh P. N. Rao, and Michael Van Wie. Hierarchical learning of reactive behaviors in an autonomous mobile robot. In *Proceedings of the IEEE International Conf. on Systems, Man, and Cybernetics*, 1995.

[11] G.E. Hinton, J.L. McClelland, and D.E. Rumelhart. Distributed representations. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1. MIT Press, Cambridge, MA, 1986.

[12] U.D. Joglekar. Learning to read aloud: A neural network approach using sparse distributed memory. Technical Report 89.27, Research Institute for Advanced Computer Science, NASA Ames Research Center, 1989.

[13] Leslie P. Kaelbling. *Learning in embedded systems*. MIT Press, Cambridge, MA, 1993.

[14] P. Kanerva. *Sparse Distributed Memory*. Cambridge, MA: Bradford Books, 1988.

[15] Pentti Kanerva. Sparse distributed memory and related models. In Mohamad H. Hassoun, editor, *Associative Neural Memories*, pages 50–76. New York: Oxford University Press, 1993.

[16] Ben J. A. Krose and Marc Eecen. A self-organizing representation of sensor space for mobile robot navigation. In *IEEE/RSJ/GI International Conference on Intelligent Robots an Systems*, pages 9–14, 1994.

[17] Benjamin J. Kuipers and Yung-Tai Byun. A robust, qualitiative approach to a spatial learning mobile robot. In *SPIE Cambridge Symposium on Optical and Optoelectronic Engineering: Advances in Intelligent Robotics Systems*, November 1988.

[18] M. A. Lewis, A. H. Fagg, and A. Solidum. Genetic programming approach to the construction of a neural network for control of a walking robot. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, Nice, France, 1992.

[19] Pattie Maes and Rodney A. Brooks. Learning to coordinate behaviors. In *Proceedings of AAAI-90*, pages 796–802, 1990.

[20] Sridhar Mahadevan and Jonathan Connell. Scaling reinforcement learning to robotics by exploiting the subsumption architecture. In *Proceedings of the Eighth International Workshop on Machine Learning*, 1991.

[21] David Marr. A theory of cerebellar cortex. *J. Physiol. (London)*, 202:437–470, 1969.

[22] Maja Mataric. Integration of representation into goal-driven behavior-based robot. *IEEE Transactions on Robotics and Automation*, 8(3):304–312, 1992.

[23] James T. McIlwain. Distributed spatial coding in the superior colliculus: A review. *Visual Neuroscience*, 6:3–13, 1991.

[24] Ulrich Nehmzow and Tim Smithers. Mapbuilding using self-organizing networks in “Really Useful Robots”. In J.-A. Meyer and S. W. Wilson, editors, *From Animals to Animats 1: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pages 152–159. Cambridge, MA: MIT Press, 1991.

[25] Randal. C. Nelson. Visual homing using an associative memory. *Biological Cybernetics*, 65:281–291, 1991.

[26] Steven J. Nowlan. Maximum likelihood competitive learning. In *Advances in Neural Information Processing Systems 2*, pages 574–582. Morgan Kaufmann, 1990.

[27] David Pierce and Benjamin Kuipers. Learning hill-climbing functions as a strategy for generating behaviors in a mobile robot. In *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pages 327–336, 1991.

[28] T. Poggio and F. Girosi. Networks for approximation and learning. *Proc. IEEE*, 78:1481–1497, 1990.

[29] D.A. Pomerleau. ALVINN: An autonomous land vehicle in a neural network. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1, pages 305–313. Morgan Kaufmann, San Mateo, 1989.

[30] D.A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, Spring 1991.

[31] R.W. Prager and F. Fallside. The modified Kanerva model for automatic speech recognition. *Computer Speech and Language*, 3(1):61–81, 1989.

[32] Rajesh P.N. Rao and Dana H. Ballard. Learning saccadic eye movements using multiscale spatial filters. In G. Tesauro, D.S. Touretzky, and T.K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 893–900. Cambridge, MA: MIT Press, 1995.

[33] Rajesh P.N. Rao and Dana H. Ballard. Natural basis functions and topographic memory for face recognition. In *Proc. of IJCAI*, pages 10–17, 1995.

[34] Rajesh P.N. Rao and Dana H. Ballard. Object indexing using an iconic sparse distributed memory. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 24–31, 1995.

[35] Rajesh P.N. Rao and Olac Fuentes. Perceptual homing by an autonomous mobile robot using sparse self-organizing sensory-motor maps. In *Proceedings of World Congress on Neural Networks (WCNN)*, pages II380–II383, 1995.

[36] David Rogers. Predicting weather using a genetic memory: A combination of Kanerva’s sparse distributed memory and Holland’s genetic algorithms. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 455–464. Morgan Kaufmann, 1990.

[37] Jun Tani and Naohiro Fukumura. Learning goal-directed sensory-based navigation of a mobile robot. *Neural Networks*, 7(3):553–563, 1994.

[38] Steven D. Whitehead and Dana H. Ballard. Learning to perceive and act by trial and error. *Machine Learning*, 7(1):45–83, 1991. (Also Tech. Report # 331, Department of Computer Science, University of Rochester, 1990.).

[39] Lambert Wixson. Scaling reinforcement learning techniques via modularity. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 368–372. Morgan Kaufmann, 1991.

[40] E. Yair, K. Zeger, and A. Gersho. Competitive learning and soft competition for vector quantizer design. *IEEE Trans. Signal Processing*, 40(2):294–309, 1992.