Application of biological learning theories to mobile robot avoidance and approach behaviors

Carolina Chang and Paolo Gaudiano †
Boston University Neurobotics Lab
Department of Cognitive and Neural Systems
677 Beacon Street
Boston, MA 02215, USA

(Received 27 May 1998)

ABSTRACT. We present a neural network that learns to control approach and avoidance behaviors in a mobile robot based on a form of animal learning known as operant conditioning. Learning, which requires no supervision, takes place as the robot moves around an environment cluttered with obstacles and light sources. The neural network requires no knowledge of the geometry of the robot or of the quality, number, or configuration of the robot's sensors. In this article we provide a detailed presentation of the model, and show our results with the Khepera and Pioneer 1 mobile robots.

KEYWORDS: Robot Learning; Operant Conditioning; Neural Networks; Obstacle Avoidance; Approach Behavior.

† We would like to thank Eduardo Zalama for his contribution to earlier versions of this work, and William Streilein for his comments on an earlier draft of this paper. Support for this research was provided by the mediaCenter mbH in Friedrichshafen, Germany, and by grants from the Office of Naval Research and DARPA (N00014-96-1-0772 and N00014-95-1-0409).
1. Introduction

When an animal has to survive in a complex, unknown environment, it must somehow learn to recognize informative cues in the environment, and to predict the consequences of its own actions. Biological organisms are a clear example that this sort of learning is possible in spite of what, from an engineering standpoint, seem to be insurmountable difficulties: noisy sensors, unknown kinematics and dynamics, nonstationary statistics, and so on.

We are interested in understanding how animals are able to solve complex problems such as learning to navigate in an unknown environment, so that we may apply what is learned from biology to the control of robots. In particular, in this article we describe a neural network model of classical and operant conditioning that learns to control the avoidance and approach behaviors of a wheeled mobile robot.

The neural network that we describe here is based on a theoretical model of classical and operant conditioning first proposed by Grossberg in 1971 (Grossberg, 1971; Grossberg, 1982). The model shows how an organism, in this case a robot, can learn without supervision to recognize simple stimuli in its environment and to associate them with different actions. In particular, our model is trained by letting a robot move around in an environment containing some objects that lead to punishment (obstacles with which the robot collides) and some other objects that lead to reward (lights). Briefly stated, whenever the robot receives punishment because of a collision, an inhibitory association is learned between the neurons representing the robot’s sensors and the neurons representing the robot’s movements. After training, a given pattern of sensor activations will tend to suppress movements that would yield punishment. Similarly, an excitatory association can be learned when the robot receives a reward (e.g., sufficiently high light intensity), so that the robot will tend to promote movements toward light sources.

Avoiding obstacles and approaching light sources is not a new achievement, nor is the application of learning theories to these problems. We believe that our approach contains several novel aspects, and that it is useful and powerful for several reasons. First, our model can learn approach and avoidance behaviors simultaneously and quite rapidly. Second, it is based on an egocentric frame of reference, so that learning in one environment generalizes to any environment. But the most important feature of our model, we believe, is that it requires no implicit or explicit knowledge about the shape of the robot, the quality and sensitivity of the sensors, or the configuration of the sensors on the robot. Hence, our model minimizes the need for calibration, and it can be of great use in applications where multiple robot platforms may be used, or where the characteristics
of the sensors are unknown or variable. To demonstrate this point we have used exactly the same network to learn approach and avoidance behaviors in two very different mobile robot platforms: the Khepera, and the Pioneer 1.

Preliminary partial results of our work have been presented in condensed form at recent meetings (Gaudiano et al., 1996; Chang and Gaudiano, 1997; Gaudiano and Chang, 1997). In this article we present a complete, detailed description of the model and of the results obtained with real robots. We begin the presentation with a brief introduction to two forms of animal learning known as classical and operant conditioning, and with a general description of the theory on which our model is based. Section 4 presents our model and results when only an avoidance behavior is learned. In Sec. 5, the model is modified to learn a light approach behavior. The model and results are extended to include simultaneous learning of approach and avoidance behaviors in Sec. 6, and to add more dynamics to the model, in Sec. 7. We close the article with discussion and conclusion sections.

2. Classical and Operant Conditioning

Psychologists have identified classical and operant conditioning as two primary forms of learning that enable animals to acquire the causal structure of their environment. In the classical conditioning paradigm, learning occurs by repeated association of a conditioned stimulus (CS), which normally has no particular significance for an animal, with an unconditioned stimulus (UCS), which has significance for an animal and always gives rise to an unconditioned response (UCR). For example, a rat that is repeatedly shocked (UCS) shortly after a red light is turned on (CS) will associate the red light with fear (UCR), meaning that eventually, presentation of the red light alone elicits a conditioned response (CR) resembling the fear response elicited by the shock itself. Hence, classical conditioning is the putative learning process that enables animals to recognize informative stimuli in the environment.

In the case of operant conditioning, an animal learns the consequences of its own actions. More specifically, the animal learns to exhibit more frequently a behavior that has led to reward in the past, and to exhibit less frequently a behavior that has led to punishment. For example, a pigeon can be trained to peck at an illuminated key in order to receive a small food reward, while a human being might learn to stop at a red light in order to avoid getting in an accident.

In the field of neural networks research, it is often suggested that neural networks based on associative learning laws can model the mechanisms of classical conditioning, while neural networks based on reinforcement learning laws can model the mechanisms of operant conditioning (Sutton and Barto, 1981;
Sutton and Barto, 1990). However, both of these classes of models are too simple to function in realistic, unstructured environments. This is not to say that associative learning and reinforcement learning do not exist in some form or another in biological organisms. Instead, the problem seems to lie in the use of rather simple, “monolithic” networks designed around each particular neural network law. At least two fundamental problems arise from these sorts of neural networks: first, the majority of neural networks function only as long as the inputs and outputs are controlled and timed carefully with respect to each other; second, most neural networks have no means of learning to discriminate “good” inputs from “bad” inputs on the basis of an internal value system.

The first of these problems has been aptly dubbed the *synchronization problem* (Grossberg, 1971; Grossberg, 1982): how can learning between a CS and a UCS occur reliably even though they are presented at different times on different trials? The problem of discriminating “good” from “bad” can be discussed in the context of *motivation*, the internal force that produces actions on the basis of the momentary balance between our needs and the demands of our environment (Dorman and Gaudiano, 1994); somehow humans and animals are able to estimate the affective value of different stimuli, and learning is constrained to those cues and events that are affectively meaningful to them.

The ability to identify and discriminate what is good from what is bad is essential for an organism to survive in an unstructured environment. In practice, neural networks are rarely left to fend for themselves in the real world, learning to recognize which things are good and which are bad. Our work demonstrates that this sort of autonomy can be achieved, at least in part, with neural models that are rooted in behavioral and physiological studies.

3. Controlling a mobile robot through operant conditioning

In 1971, Grossberg proposed a detailed neural network theory of classical and operant conditioning which was designed to account for a variety of behavioral data on learning in vertebrates. The model was refined in several subsequent publications. Grossberg and Levine (1987), and Grossberg and Schmajuk (1987) report on detailed computer simulations of different components of the conditioning circuit.

Before providing details of the model and our own implementation of it, we provide an intuitive description of the main elements of the model. Figure 1 is a schematic of the overall structure of Grossberg’s conditioning theory. In the figure, populations of neurons are represented by boxes, while the interconnections between populations are represented by lines. We use the term “population” to
refer to a collection of simulated neurons performing a given function; this is comparable to the term “layer” used in other contexts.

The essential departure from a typical associative memory model is in the use of motivational signals to modulate learning. At the core of the model are several assumptions, which Grossberg (1982) describes in terms of psychological postulates. The first design consideration of the model is that those stimuli that are initially not significant to the organism (i.e., CSs) are unable to generate emotional or behavioral responses, whereas a few stimuli that are innately significant to the organism (i.e., UCSs) always lead to an emotional and behavioral response (UCR). This is represented in Fig. 1 by the modifiable connections (indicated by semi-circles) between the CS population and the Reward/Punishment population, and by modifiable connections between the Gated CS population and the Behavior Generation population. In contrast, the UCS operates through fixed, strong connections to these populations, which are represented by thick arrows in the figure. The gated CS nodes require joint activation of the sensory (i.e., CS) and emotional (i.e., Reward/Punishment) input in order to be activated. Hence, prior to learning, as long as the connections from the CS to the Reward/Punishment population are weak, the gated CS nodes cannot be activated by a CS alone, and behaviors cannot be generated by the CS.

Another psychological postulate states that a CS can learn to generate emotional and behavioral responses on its own by repeated pairing with a UCS. To satisfy this criterion, each UCS activates two populations: the Reward/Punishment population and the Behavior Generation population. The nodes of the Re-
ward/Punishment population, which Grossberg refers to as drive nodes, carry the emotional valence of the UCS. For instance, shock is a UCS that elicits fear, while food is a UCS that elicits pleasure. In a similar fashion, different UCS stimuli can generate different behaviors: shock generates an avoidance behavior, while food generates an approach behavior.

Through repeated pairing with a UCS, a CS can acquire the ability to generate emotional and behavioral responses that resemble those of the UCS with which it is paired. So, for instance, a bell that is repeatedly paired with shock will eventually elicit fear and avoidance behavior when presented alone, while a light repeatedly paired with the arrival of food will eventually elicit pleasure and approach behavior (as Pavlov’s dogs learned to salivate in response to the ticking sound of a metronome).

In summary, the Reward/Punishment (or drive) nodes restrict learning to stimuli that are paired with emotionally significant events. This is an important departure from traditional connectionist approaches where every input-output pair presented to the network is learned.

4. Learning obstacle avoidance behavior

In this section we begin with a description of the conditioning model when the entire Reward/Punishment population consists of a single node. This results in a network that can only learn to generate one kind of behavior; in our case, we use this simple version of the network to learn obstacle avoidance. In a later section we extend the model to allow simultaneous learning of both approach and avoidance behaviors.

Figure 2 illustrates our detailed implementation of the circuit schematized in Fig. 1. Each block in the diagram is replaced by a more detailed representation of the corresponding neural population. In this model the sensory cues (i.e., CSs) are stored in Short Term Memory (STM) within the population labeled S. This population includes competitive interactions to ensure that the most salient cues are contrast-enhanced and stored in STM while less salient cues are suppressed. In the present model the CS nodes correspond to activation from the robot’s range sensors.

The drive node D corresponds to the Reward/Punishment component of Fig. 1. Learning can only occur when the drive node is active. The cells in population P correspond to the Gated Conditioned Stimuli, and are represented as triangular nodes to denote that they are polyvalent cells. Polyvalent cells require the convergence of two types of input in order to become active. As described in the schematic model, these inputs come from the CS population and from the Reward/Punishment (i.e., drive) node.
Figure 2. Conditioning model for obstacle avoidance. The robot's sensor activities represent the CSs. A collision detector activates the UCS. Motor learning occurs at a population coding the robot's target angular velocity. After conditioning, the pattern of activity across the range sensors can predict a collision and modify the robot's angular velocity to avoid obstacles.

According to Grossberg’s theory, the drive node is also polyvalent: it needs the joint activation of a stimulus, and an internal homeostatic signal in order to become active. An example of a homeostatic signal is hunger, which indicates the body’s internal need for food when the body detects a low concentration of sugar in the bloodstream. In this case, an animal will not eat even in the presence of food unless it is hungry. A different situation arises in the case of aversive stimuli: an animal should always perform an avoidance behavior in the presence of an aversive stimulus. One way to interpret this is that there is a homeostatic signal corresponding to a sort of “survival instinct”, which is active at all times. In our model, the UCS corresponds to the robot colliding with an obstacle. Assuming that the survival instinct signal is always on, the drive node associated with aversive stimuli (and thus with avoidance behaviors) only requires a collision signal in order to become active. The collision signal can be generated by a bump sensor, or by any one of the proximity detectors reaching its maximum value, or when the wheels stop moving unexpectedly.

Finally, the neurons at the far right of Fig. 2 represent the network’s responses (conditioned or unconditioned), and are thus connected to the motor system. In a normal organism there may be many such networks, some giving rise to emotional responses (e.g., changes in skin conductance) and others generating
actual motor behaviors. In our model the responses are generated as a range of angular velocities that drive the robot’s movements.

We now turn to a detailed description of the model shown in Fig. 2.

4.1. DESCRIPTION OF THE MODEL’S DYNAMICS

The design principle behind Fig. 2 is that whenever the robot collides with an obstacle, learning in the circuit will tend to suppress the behavior that was being exhibited by the robot at the time of collision. This suppression of a behavior is achieved by modifiable inhibitory connections between the pattern of sensor activity and the angular velocity of the robot when the collision took place. After learning, sensor activity will lead to inhibition of those angular velocities that previously caused collision under similar sensor pattern activation. In other words, because collisions correspond to punishment, the network learns to decrease the occurrence of actions that lead to punishment, as with the typical operant conditioning paradigm.

While it moves, the robot takes measurements continuously from its range sensors. Contrast enhancement enables those sensors detecting closer objects to activate more strongly their corresponding nodes at the sensory population $S$. Originally, the $S$ population was modeled by Grossberg as a recurrent competitive field, which removes noise while contrast enhancing the input pattern (Grossberg, 1971; Grossberg, 1982). In our implementation, we have replaced the competition of activations $x_{1i}$ of population $S$ by a simple normalization rule:

$$x_{1i}(t) = \frac{I_i(t)}{\sum_j I_j(t)}$$  (4.1)

Here $I_i$ represents a sensor value which codes proximal objects with large values, and distal objects with small values. For instance, $I_i$ corresponds to “raw” measurements of infrared sensors, while it corresponds to the complement of the raw measurements (i.e., maximum range minus the actual measurement) when ultrasound sensors are used. This is because infrared returns are larger for closer objects, while ultrasound returns are smaller for closer objects.

Notice that this is the only consideration we have to make for the network to work with different types of sensors. The network requires no knowledge of the geometry of the robot or the quality, number, or distribution of sensors over the robot’s body.

The activation of drive node $D$ is determined by the weighted sum of all the CS inputs, plus the UCS input, which is presumed to have a large, fixed connection strength:
\[ y(t) = \sum_{i} x_{1i} z_{2i}(t) - T_y + UCS(t) \] (4.2)

where \( y(t) \) is the activation of the drive node, \( z_{2i} \) is the adaptive weight connecting the sensory node \( x_{1i} \) to the drive node, \( T_y \) is a threshold that controls how easily the drive node is activated, and \( UCS(t) \) represents the collision status at time \( t \) \((UCS = 1 \text{ if a collision just occurred, and } UCS = 0 \text{ otherwise})\).

The activation of the drive node and of the sensory nodes converges upon the population of polyvalent cells. In particular each polyvalent cell receives input from only one sensory node, and all polyvalent cells also receive input from the drive node. Denoting by \( x_{2i} \) the activation of the \( i \)th polyvalent cell:

\[ x_{2i}(t) = x_{1i}(t) f(y(t)) \] (4.3)

where \( f(y(t)) \) is defined as:

\[ f(y(t)) = \begin{cases} 
1 & \text{if } y(t) > 0 \\
0 & \text{otherwise}
\end{cases} \] (4.4)

The multiplication of \( x_{1i}(t) \) and \( f(y(t)) \) in Eq. 4.3 codes the need for joint activation from the sensory nodes (CS) and the drive node (Punishment/Reward), in order for the gated CSs to become active, as explained in Sec. 3. The above equations capture, in a simplified form, the dynamics of the nodes in the network. We now describe the mechanisms that lead to learning.

### 4.2. Learning to Suppress Harmful Behaviors

Let us consider a simple example in which a single sensory node is active. Before a collision occurs, and before any learning has taken place, the CS node sends activation to its corresponding polyvalent cell as given by Eq. 4.3. However, the connection from the CS node to the drive node is very weak, so that the drive node does not activate (Eq. 4.2). Hence the polyvalent cell only receives one kind of input, and it does not become active.

When the same CS node is on at the time of a collision, the UCS causes the drive node to become active. The drive node sends its activation to all polyvalent cells; however, only the polyvalent cell corresponding to the active CS turns on, because it is the only one receiving both kinds of input. At this point, two kinds of learning can take place simultaneously. First, the active CS node has the opportunity to increase the strength \( z_{1i} \) of its connection to the drive node, because they are simultaneously active. Second, the polyvalent cell corresponding...
to the CS can learn to suppress the robot’s behavior taking place at the time of the collision (details below). Each time that the same CS-UCS pairing takes place, the connection between the CS node and the drive node is strengthened; at the same time, the polyvalent cell learns more and more to suppress the behavior causing the collision. Eventually, turning on the CS node is sufficient to also turn on the drive node, even in the absence of a collision signal. As the drive node is activated by the CS, the polyvalent cell also becomes active, and it suppresses the behavior that would have caused the collision, before the collision actually takes place. Hence the neural network has learned to avoid obstacles.

In order to complete our explanation, we now describe the laws governing these two forms of learning, and we describe the function of the population that is actually controlling the robot’s behaviors.

The first type of learning follows a simple associative learning law with decay: the connection from the CS to the drive node should increase if the two are active simultaneously, but it should also decrease gradually with time, so that eventually an association can be forgotten if it is no longer useful. This learning law is given by:

\[ z_{1i}(t) = Ez_{1i}(t-1) + Fx_{1i}(t)f(y(t)) \]  

(4.5)

where \( E \) is the weight decay rate, and \( F \) is the learning rate when CS and UCS are simultaneously active.

The second type of learning, taking place between the polyvalent cells and the cells that actually generate the robot’s movements, is a form of inhibitory learning, whereby simultaneous activation of the pre- and post-synaptic cells leads to an increasingly large negative (i.e., inhibitory) weight. We adopt this inhibitory law to capture the idea that, in the case of a punishing UCS, the behavior that led to the UCS will be suppressed. Before describing the inhibitory learning law, we must explain the scheme that we use to generate the robot’s behaviors.

### 4.3. The Angular Velocity Map

Fig. 3 illustrates the scheme we used to control the movements of the robot. We use a 1-D population of neurons that represent the robot’s angular velocity: activating the leftmost node causes the robot to turn left at its maximum angular velocity \((-w_m \frac{rad}{s})\); activating the rightmost node causes the robot to turn right at its maximum angular velocity \((+w_m \frac{rad}{s})\); activating the central node causes the robot to move straight ahead. For the results reported here, the activity is distributed over the entire angular velocity map, but at any given time the
robot’s angular velocity is selected on the basis of the maximally active node in the map.

Figure 3 shows that the spacing between nodes in the angular velocity map is not uniform. This is meant to capture the idea that the distribution of angular velocities is not uniform, being more densely sampled in the center than at the sides of the angular velocity map. We do this so that shifting activation, for example, from the central node (straight ahead) to the next node on the right, leads to only a small change in angular velocity, while shifting from the leftmost node to the next node to its right leads to a larger change in angular velocity. This nonuniform distribution of angular velocities yields smooth robot movements even when using a modest number of neurons (we use 11 in all our results).

Let us consider a situation in which no obstacles are in the environment, and thus no sensory nodes are active. Hence we can momentarily ignore the conditioning circuit to focus on the function of the angular velocity map. Let’s say that we want the robot to move toward a target to its left. We presume that the greater is the angle through which the robot must turn, the greater the angular velocity that will be selected. The particular “desired” node $n_d$ in the angular velocity map that should be turned on in order to generate an angular velocity $\alpha$ is given by:
Figure 4. The peak shift property. The positive Gaussian distribution represents the desired angular velocity, whereas the negative distribution represents the activation from the conditioning circuit. The summation of the two distributions determines the angular velocity that will be used to drive the robot. Notice how the peak of the excitatory Gaussian is shifted by the inhibitory Gaussian.

\[
n_d(t) = \begin{cases} 
\frac{N}{2} + \frac{N(\alpha_w + 0.5 w_m) \alpha(t)}{w_m(1.8 + \alpha(t))} & \text{if } \alpha(t) > 0 \\
\frac{N}{2} + \frac{N(\alpha_w + 0.5 w_m) \alpha(t)}{w_m(1.8 - \alpha(t))} & \text{otherwise}
\end{cases}
\]  

(4.6)

where \( N \) is the number of nodes in the angular velocity map, \( w_m \) is the maximum angular velocity, and \( \alpha_w \) is a constant that controls the slope of the sigmoid function. This equation is simply a sigmoidal function of the type shown in Fig. 3, which is how we achieve a non-uniform distribution of angular velocities as described above.

Equation 4.6 selects a single node corresponding to the desired angular velocity. However, in order to achieve smooth changes in velocity, we actually turn on a Gaussian distribution of activity centered around the node \( n_d \):

\[
gx_j(t) = e^{-(j - n_d(t))^2 / \sigma^2}
\]

(4.7)

where \( \sigma \) is the standard deviation of the Gaussian. Any reasonable choice of \( \sigma \) (say, about one third of the number of nodes in the map) yields satisfactory results.
The use of a Gaussian is crucial for our obstacle avoidance scheme, as depicted in Fig. 4. We have taken advantage of a nice property of Gaussians (actually any bump-like function works equally well): When an excitatory Gaussian is combined with an inhibitory Gaussian at a slightly shifted position, the resulting net pattern of activity exhibits a maximum peak that is shifted from the excitatory Gaussian in a direction away from the peak of the inhibitory Gaussian. For instance, in Fig. 4 we superimpose a positive Gaussian centered over the middle of the map, and a negative Gaussian to the right. The net pattern of activation shows a positive peak that is shifted to the left of center.

Imagine now that the positive Gaussian represents the angular velocity at which the robot must move to reach a target (straight ahead), and that the negative Gaussian represents an obstacle to the right of the robot. In this case, the two Gaussians interact, causing the robot to turn to the left, avoiding the obstacle. This peak shift property has several desirable features, such as the fact that the shift becomes greater as the peaks of the Gaussians become closer (Gaudiano et al., 1996b), and is thus well suited as the basis for obstacle avoidance. We now conclude our explanation of the obstacle avoidance model by explaining how the polyvalent cells learn to generate the “correct” inhibitory Gaussian that will make the robot turn away from obstacles.

### 4.3.1. The inhibitory learning law

By using an inhibitory learning law, the polyvalent cells corresponding to the active sensory nodes acquire negative connection weights that learn to generate a pattern of inhibition matching the angular velocity profile active at the time of collision. For instance, if the robot was turning right and collided with an obstacle, the range sensor neuron most active shortly before the collision will learn to generate an inhibitory Gaussian centered upon the right-turn node in the angular velocity population. The learning law that leads to the development of negative weights is given by:

\[
zm_{i,j}(t) = zm_{i,j}(t-1) - M x_{2i}(t) \left[ \frac{g x_j(t)}{1 + (j - G(t))^2} + zm_{i,j}(t-1) \right] \tag{4.8}
\]

where \(zm_{i,j}\) represents the adaptive weight from the polyvalent cell \(i\) to the node \(j\) of the angular velocity map. \(M\) is the learning rate, \(g x_j\) is the Gaussian function centered on the desired movement direction, as given above in Eq. 4.7, \(G(t)\) is the index of the node in the angular velocity map for which \(g x_j\) is maximal. This learning law simply generates an inhibitory Gaussian centered around the node that was most active at the time of the collision.
4.4. Activation of the Angular Velocity Map

Once learning has occurred, the activation of the angular velocity map is given by two components (Fig. 4). An excitatory component reflects the angular velocity required to reach a given target in the absence of obstacles. We have shown previously how this signal can be derived from the sensors (Gaudiano et al., 1996b); for simplicity here we assume that the angular velocity is proportional to the angle between the robot’s current heading and the target. A second, inhibitory component, generated by the conditioning model in response to sensed obstacles, moves the robot away from the obstacles as a result of the activation of sensory signals in the conditioning circuit. The equation that describes this behavior is:

\[ x m_j(t) = g x_j(t) + \sum_i x_{2i}(t) z m_{i,j}(t - 1) \]  

(4.9)

The node in the angular velocity map that has maximal activation after the summation of the excitatory and inhibitory Gaussians determines the angular velocity that the robot will perform in its next movement. In particular, given that after summing the excitatory and inhibitory terms, node J is the most active node at time \( t \), the angular velocity used to drive the robot is given by:

\[ w(t) = \begin{cases} \frac{w_m[J(t) - N/2]}{N(\alpha w + 0.5w_m) - w_m[J(t) - N/2]} & \text{if } J(t) > N/2 \\ \frac{w_m[J(t) - N/2]}{N(\alpha w + 0.5w_m) + w_m[J(t) - N/2]} & \text{otherwise} \end{cases} \]  

(4.10)

Notice that Eq. 4.10 is simply the inverse function of Eq. 4.6. In other words, Eq. 4.6 was used to explain how a given angular velocity is mapped onto the population, while Eq. 4.10 shows how the activation of a given node is actually used to generate the angular velocity.

After learning, when the robot is required to reach a target by turning at a certain angular velocity \( \alpha \), the actual angular velocity might differ if obstacles are detected: in the presence of obstacles, the polyvalent cells generate inhibitory activation, causing the peak in the angular velocity map to shift, moving the robot away from obstacles, and also away from its instantaneous desired direction. Once the obstacle has passed, the robot’s movements will simply be determined by the “desired” angular velocity.

The output of the angular velocity population is decomposed algorithmically into left and right wheel angular velocities. In an alternative approach by Gau-
diano et al. (1996b), the transformation from the angular velocity population to actual wheel velocities can be done adaptively with another neural network.

The technique we have described for obstacle avoidance (i.e., using a difference of Gaussians) is related to the technique widely known as potential fields (Khatib, 1986; Latombe, 1991), though the methods differ in various details. For instance, we only utilize a one-dimensional map of neurons representing instantaneous desired angular velocities, rather than actually building a potential function based on sensor activities. Nonetheless, the approach used here is similar to potential fields and other methods that “weight” the presence of obstacles sensed around the robot. In fact, this part of the circuit should be easy to replace with an alternative, but comparable method. The present method has the desirable features of being computationally expedient, easy to implement, and robust to parameter manipulation. We also chose this particular technique because of our prior experience with it (Gaudiano et al., 1996b; Muñiz et al., 1995; Zalama et al., 1995).

4.5. Experimental results

We have implemented the model just described on two real mobile robots. The Pioneer 1 (Real World Interface, Jaffrey, NH), shown in Fig. 5(a), is a small (14" wide, 18" long, 9" tall), two-wheel differential-drive robot with five forward-facing and two side-facing sonar range finders. The Khepera (K-team SA, Préverenges, Switzerland), shown in Fig. 5(b), is a miniature (2.2" diameter) differential-drive robot with eight infrared proximity sensors, six of which cover the frontal 180°, and the remaining two sensors cover the back of the robot. In our experiments we have ignored the two rear-facing infrareds, using only the six frontal sensors. We have previously reported our results using simulators (Gaudiano et al., 1996a; Chang and Gaudiano, 1997). We focus here on the results using real robots.

It is worthwhile to note that the implemented model requires essentially no modifications in order to run on these two robots, the only difference being that the infrared sensors on the Khepera return larger values for closer objects, while the sonars on the Pioneer 1 return smaller values for closer objects (of course, there is also a difference in the number of CS nodes).

In our model, the range sensors initially do not propagate activity to the motor population because the initial weights are small or zero. The robot is trained by allowing it to make random movements in a cluttered environment. The goal of the training phase is to give each CS node the opportunity to sample several movements that lead to collisions. In practice we found that it is sufficient for each CS node to be active during only a handful of collisions, when using 11 nodes in the angular velocity map. In order to generate a wide range of movements,
Figure 5. Two robotics platforms. (a) The Pioneer 1 robot; (b) the Khepera robot.

Figure 6. (a) Learning in the Khepera robot, measured as the number of collisions in 500 steps as a function of the total number of collisions experienced during training. Min and Max refer to the best and worst learning curves out of a set of five training trials. (b) Adaptive connections between the sensors and the angular velocity map developed by the Khepera robot for the obstacle avoidance behavior.

during the training phase we turn on each node in the angular map for a brief time until a collision is registered, then switch to a new angular map node and repeat the process. We can achieve good avoidance behavior in this way with only a few collisions for each node.

Figure 6(a) illustrates the learning process. We obtained this curve in the
following way: starting with all the weights in the network set to zero, we turn on one node in the angular map and let the robot move until it collides with an obstacle, generating a small amount of learning. We then turn on another node, repeat the process, and so on. At regular intervals during the training phase we temporarily disable learning and allow the robot to move from a new starting position for a total of 500 steps through the cluttered environment, and measure for how many of the 500 steps the robot detected a collision. On the first trial, before any learning has taken place, as soon as the robot collides it remains stuck against the obstacle, so the number of collisions is very close to 500. By the time we have trained through 50 collisions (total: meaning that each of the six sensors, on average, has sampled fewer than ten collisions) the robot is able to navigate with virtually no collisions.
The inhibitory weights developed by the neural network are depicted in Fig. 6(b). The adaptive connections between the sensory nodes and the angular velocity map develop in such a way that angular velocities that make the robot turn to the right (nodes close to 10) are inhibited when the sensors located at the right side of the robot are active (sensory nodes 4 and 5). Similar yet opposite inhibitory weights develop for left turns when obstacles are sensed at the left side. In the middle of the figure (nearly straight-ahead movements with obstacles located straight ahead), a Gaussian-like inhibitory curve accounts for the fact that in such cases, turns to either the left or the right are needed to avoid collisions.

After sufficient training, the robot is able to wander in a cluttered and constantly changing environment while avoiding collisions with obstacles. Fig. 7(b) and (c) are digital images captured using a frame grabber board that receives signals from a camcorder mounted above the Khepera’s environment (a). Miniature furniture was placed to simulate a house environment. A tracking algorithm localizes the robot’s position and direction (square) and traces the trajectory described by the robot (black dots). The target position, which is represented by a black circle, is ignored in the wander task and approached in the target

Figure 8. Obstacle avoidance behavior of the Pioneer 1 robot. (a) An obstacle is located in the robot’s current direction of movement. (b) The robot makes a turn to the left to avoid a collision with the obstacle. (c) The robot has circumnavigated the obstacle without colliding with it.
reaching task. The robot makes wide and fast turns when it gets very close to the obstacle due to the short range of activation of the infrared sensors (the range of the Khepera’s IR sensors can vary, but in our environment it is limited to approximately 1 inch). Nonetheless, the resulting movements succeed in preventing collisions. The robot is also capable of reactive target reaching (c). Without having a map of the environment, in most cases the robot is able to reach arbitrary target positions, when the angle and the distance between the current position and the goal are specified. Appropriate nodes of the angular velocity map activate depending on the angle between the robot’s heading direction and the target. As it travels, the robot updates its position and direction by relying on its odometry. The learned inhibitory Gaussians force the robot to deviate from its desired trajectory when the proximity of objects is sensed.

The same kind of results were obtained when we trained the Pioneer 1 robot to develop the obstacle avoidance behavior. About 60 collisions where required during the training phase to fully learn to avoid obstacles, again corresponding to less than ten collisions per sensor (given the Pioneer’s total of seven sonars). Fig. 8 shows some images of the Pioneer 1 moving in our Lab while avoiding obstacles. A main difference between the two robots is that the Pioneer 1 robot developed more gradual avoidance movements than the Khepera. This happened because its ultrasound sensors are accurate in the range of several feet, whereas the Khepera’s infrared sensors can only detect objects in the range of 1 inch.

5. Learning to generate approach behaviors

The neural network model described in the previous section has been used to develop an obstacle avoidance behavior in our robots. Learning of the inhibitory Gaussian distribution in order to produce a peak shift in the activation of the angular velocity map leads to the desired avoidance behavior.

In another experiment, we used the same neural network design of Fig. 2 in order to develop an approach behavior in the robot. Notice that this behavior is completely opposite to the obstacle avoidance behavior since it requires the robot to move towards the location of the source of sensory stimulation (instead of away).

In this case, detection of an increase in the level of light present in the environment corresponds to the UCS. For the CSs, we used the ambient light intensity measurements of the Khepera’s infrared sensors. Light is regarded as a reward, e.g., food, which activates a “pleasure” drive, and elicits the approach behavior.

To generate the approach behavior, the associative learning law of Eq. 4.8 was modified in order to build excitatory Gaussian distributions that would move
the robot towards the location of sensory stimulation, as depicted in Fig. 9. The new learning law is given by:

\[ zm_{i,j}(t) = zm_{i,j}(t - 1) + Mx_{2i}(t) \left[ \frac{gx_j(t)}{1 + (j - G(t))^2} - zm_{i,j}(t - 1) \right] \]  

Equation 5.1 allows the network to develop positive weights in the connections from the polyvalent cells to the angular velocity map. Recall that previously, collision elicited punishment signals that lead to inhibition, and thus, to the obstacle avoidance behavior. With Eq. 5.1, the sensory neurons active at the time of reward form excitatory connections to the angular velocity map. As a result, activation of light-sensitive sensory neurons leads to a positive Gaussian that effectively shifts the target Gaussian in the direction of the sensory (light) stimulation.

5.1. Experimental results

As in the case of obstacle avoidance, after training the robot successfully learned to turn towards the source of light. If the light is moved, the approach behavior enables the robot to follow the trajectory of the light, as shown in Fig. 10.

Figure 11 depicts the excitatory weights developed by the network for the light
approach behavior. Angular velocities nodes that make the robot turn right (angular velocity nodes close to 10) are excited when the sensors located at the right side of the robot (sensory nodes 4 and 5) are active. A similar, mirror-symmetrical pattern of weights develop for the left side of the robot. In the middle of the figure, a Gaussian-like shape indicates that straight-forward movements are favored when light sources are located straight ahead of the robot. The adaptive connections developed in the approach behavior are in many ways similar to the weights developed in the obstacle avoidance behavior (Fig. 6(b)). Notice however that in Fig. 11 the weights are excitatory whereas they are inhibitory in the obstacle avoidance behavior. Although the weights seem to grow faster for extreme angular velocities in the light approach behavior, the relation between sensory stimulation and the required turn is clear in both behaviors.

In the light approach behavior the robot was instructed to stop when any of its two front sensors became 80% or more active. Although this stopping mechanism used explicit information about the location of the sensor, notice that as in the obstacle avoidance case, the acquisition of the approach behavior made no use of this or any other information about the robot’s geometry.

How close the robot gets to the source of light is determined by the characteristics of the sensor and the light intensity. With a light source of 1 Watt the sensors of the Khepera become 80% active at a distance of approximately 2.5”.

In our experiments, a 2.2V flashlight was used. The top view of the environment (Fig. 12(a)) shows the robot at the left side of the figure. The flashlight was placed 20” away from the robot, and was directed to the robot’s leftmost
Figure 11. Adaptive connections between the sensors and the angular velocity map developed by the Khepera robot for the approach behavior.

sensor. The front of the robot and the flashlight formed a 90° angle. This was the starting position for the experiments described below.

When the robot was instructed to move (Fig. 12(b)), it made a quick turn to face the light. At this time the front sensors were already very active, which caused the robot to stay at the same position, 20° from the flashlight.

To see the effect of the light source on the approach behavior, we passed the light beam through a narrow aperture, thus decreasing the amount of light reaching the sensors as well as the beam width. The flashlight, which has a width of 2", was partially occluded by two walls, separated by a variable distance (smaller than 2"). As the open space was made narrower, the robot started to approach the flashlight in a straight trajectory until any of its frontal sensors was 80% active. In the example shown in Fig. 12(c), when the wall aperture was 0.65" the robots moved 8" towards the source of light. It can be observed that as the robot approached the light, it lost track of the light, but turned again towards it and finally stopped. For even narrower wall apertures (smaller than 0.65") the robot was successful in turning towards the light but it was unable to approach it, as shown in Fig. 12(d).

From Fig. 12 (c) and (d) one might conclude that the robot over- and undershoots in its turns toward the light. Otherwise straight trajectories would be observed in all cases. This is absolutely correct. As a matter of fact, the Khepera robot has no sensors right at the front (0 degrees). Its two frontal sensors are located at 10° and -10°, separated by approximately 0.625". With the turn and approach movements, the robot effectively seeks to increase its sensor activation.
Figure 12. Light approach behavior of the Khepera robot. (a) For each trial the robot is placed at 90°, and 20° from a flashlight. (b) The robot makes a turn and faces the light. (c) Two walls with a 0.65” separation narrow the source of light. The robot turns and moves 8” towards the light. (d) For narrower separation the robot still turns, but misses the source of light.

With the narrow source of light, only one of the two frontal sensors can receive the light directly, and this requires the robot to head slightly to one side.

The Khepera configuration also explains why the robot lost the source of light in Fig. 12 (c) and (d). Due to the 0.625” of separation between the sensors, as the robot moves there are periods of time when the light is not received by the robot because it falls in the gap between any two sensors. In (c) the light was subsequently detected by a side sensor, which allowed the robot to make a second turn towards the flashlight. In (d) however, the light did not stimulate the side sensors sufficiently, which prevented the robot from approaching the light.
Finally, Fig. 13(a) shows a setup we used to illustrate how the target reaching task is partially affected by the light approach behavior. The flashlight was placed in such a way that the light would be encountered as the robot navigated to the target position (black circle). When allowed to move (Fig. 13(b)), the robot turned towards the target and started to approach it. However, as soon as the light was detected, the robot turned toward the flashlight and tried to approach it. The light approach behavior was released due to co-activation of the pleasure drive and the light sensors. However, the change in the robot’s heading direction activated the node in the angular velocity map required to reach the target position, and the robot turned again towards the target. Once far from the influence of light, the robot approached the target with no further detours. This effect of the light approach behavior in the robot’s trajectory can be better identified when compared to the straight trajectories obtained in the absence of a source of light (Fig. 13(c)).

6. Multi-Drive Network

So far we have described an implementation of the conditioning circuit that can learn to either avoid obstacles or approach the source of sensory stimulation.
As presented, the model is unable to learn both behaviors simultaneously. In order to be able to learn multiple behaviors, the original network of Fig. 2 was expanded as shown in Fig. 14.

The expanded model consists of sensory nodes for both kinds of cues (i.e., proximity of objects, and intensity of light). For simplicity, the figure shows only the connections of one node of each type. Note that in principle, there is no difference between the two types of nodes other than the kind of sensory information they are concerned with. To be able to elicit two opposite behaviors, the network contains two drive nodes, i.e., fear and pleasure. “Survival” is the internal homeostatic signal associated with fear and the avoidance behavior, while “hunger” is the signal associated with pleasure and the approach behavior. Each drive releases incentive motivation to a specific population of polyvalent cells. In the figure, Population $P-$ is associated with the fear drive and avoidance behavior, and population $P+$ is associated with the pleasure drive and approach behavior.

In the expanded conditioning circuit, the two drives compete in a sensory-drive heterarchy (Grossberg, 1971). That is to say, the combination of sensory activation and drive activation determines which drive will release incentive motivation, thus allowing an appropriate motor response to take place. For simplicity, we have assumed that both internal homeostatic signals (i.e., survival and hunger) are always active and equally strong. This reduces the problem of drive competition to a matter of sensory activity: after learning, the strongest cues will win the competition leading to the release of the associated behavior. In a
later section we also describe a modification of the appetitive (pleasure) drive to mimic a dynamically modulated hunger signal.

The population of polyvalent cells that receives incentive motivation from the winning drive node elicits the motor response with which it is associated. Hence, at each time only one motor response is released, i.e., either the avoidance or the approach behavior.

Note that when a drive node wins the competition, it releases incentive motivation to all the polyvalent cells in the corresponding population ($P+$ or $P-$). That is, each drive releases incentive motivation to the polyvalent cells connected with all the proximity sensory nodes and also to the cells connected with all the light detection sensory nodes. This feature reflects the observation that prior to learning the CSs have no special meaning; it is only through learning that the network starts to discover the causal structure of the environment. When a collision occurs or food is sighted, all sensory nodes are allowed to learn. However, after repeated associations, only sensory nodes systematically active during learning will have their connections strengthened. In our example, activation of the fear drive tends to co-occur with activation of proximity detector nodes, so that ultimately the activation of proximity detectors generates an avoidance behavior by the development of inhibitory weights to the angular velocity population, as described earlier. At the same time activation of the pleasure drive tends to co-occur with activation of ambient light detectors, which generates an approach behavior. In principle, however, we could have associated the ambient light detectors with avoidance behaviors and the proximity detectors with approach behaviors simply by changing the contingencies of the problem. For instance, imagine an environment in which all the obstacles are bright lights, and a battery charging (i.e., feeding) station is the only non-lit obstacle. Then the robots can use exactly the same circuit and exactly the same sensors to learn to avoid lights and approach the non-lit obstacle. A more interesting example might develop a wall-following behavior by activating the fear drive when the frontal proximity sensors are used, but activating the pleasure drive when the lateral proximity sensors are activated.

The two forms of learning take place simultaneously while the robot is moving through the environment. The dynamics of the network determine which nodes learn in which conditions. In general, as long as lights and obstacles are not all overlapping, there is no problem with simultaneous learning of both types of behaviors. Likewise, after learning, the expanded neural network is capable of exhibiting multiple behaviors depending on the events that take place in the environment.
Simultaneous learning of the avoidance and approach behaviors in the extended conditioning model developed quite nicely. The robot was able to exhibit the approach and avoidance behaviors depending on the moment-by-moment activation of its sensors.

Figure 15 shows the Khepera approaching light while avoiding obstacles. In part (a), navigation was guided by the approach behavior, as the robot followed a flashlight directed by the experimenter. The experimenter brought the robot close to a wall, and at that moment removed the flashlight. In part (b), the robot avoided the nearby wall and kept wandering the environment, avoiding further obstacles that it found in its path.

A more interesting result is obtained when pleasure and fear compete in the sensory-drive heterarchy. The effect of multiple drive activation is shown in Fig. 16. A similar setup to the experiment shown in Fig. 12 was used. However, a small change was introduced. Instead of having two separated walls occluding the flashlight, a long wall with a 1” wide window was used. The 1” width of the window ensured that the light would be easily detected (as described in Sec. 5.1), while the window configuration allowed the robot to detect an obstacle in the same place where it detected the source of light.

The robot was placed not far from the source of light, heading diagonally (Fig. 16(a)). There was no special reason why those initial position and direction were chosen, other than having the robot stop near the wall. In the figure, it can
be observed that when the robot detected the light, its front light sensors were already very active, and therefore, the robot stopped without needing to turn.

Then, the flashlight was moved far away from the wall but it was still directed towards the window (Fig. 16(b)). As the activation of the light sensors decreased, the robot turned towards the window and moved forward. When the robot approached the wall, its sensors started to signal the proximity of an obstacle. Some zig-zag in the robot's movement occurred due to the pleasure and fear competition as the robot tried to approach and avoid the wall at the same time. However, when the robot got very close to the wall, the proximity sensors became very active, which allowed fear to ultimately win the competition in the sensory-drive hierarchy. The robot then turned to avoid the wall and continued its navigation. In brief, competition between pleasure and fear allowed the robot to avoid obstacles even when approaching a source of light.

7. Adding dynamics to the drive activation cycle

In the case of obstacle avoidance, it is reasonable to assume the existence of a survival instinct that is always active: the robot should never hit the walls. On the other hand, if we equate light approach with feeding, it is not reasonable to assume that the robot is always hungry. For this reason, we modified the proposed network in order to account for a more natural homeostatic signal activation. We wanted the hunger homeostatic signal to become inactive after
food intake, and to activate again after a variable period of time, which depends on the size of the last food intake.

To this end, we employed gated dipoles (Grossberg, 1982). In brief, the gated dipole is a neural circuit consisting of two channels organized in a mutually inhibitory, or opponent fashion (Fig. 17). When an input is turned on, and subsequently turned off, the first channel, called the on-channel, generates a sustained but habituating response (e.g., fear) to the onset of a stimulus (e.g., shock). In contrast, the off-channel produces a transient response (e.g., relief) to the offset of the same cue. The activation of the off-channel due to the stimulus offset is known as antagonistic rebound. A detailed description of the gated dipole, which is beyond the scope of this paper, can be found elsewhere (Grossberg, 1972; Grossberg, 1982).

To implement the "hunger" homeostatic signal, two coupled gated dipoles were used (Fig. 18). One gated dipole describes the amount of "food" (e.g., light) that the robot has consumed. Positive activation represents the time the robot is eating while non-positive values indicates the time the robot is not eating. This information is used by a second, slower gated dipole to control the onset/offset
Figure 18. Two coupled gated dipoles were integrated into the extended conditioning model in order to describe the dynamics of the “hunger” homeostatic signal.

of the hunger homeostatic signal. A period of time without eating, i.e., when the activation of the first dipole is close to zero, triggers the hunger signal, which remains on until food has been consumed (at which time the first dipole’s on-channel activation changes from positive to negative). At that moment, the hunger signal switches off, and “satiation” occurs (i.e., negative activation of the on-channel of the second dipole). Satiation decreases with time, eventually leading to reactivation of the hunger signal.

Co-activation of the hunger homeostatic signal and conditioned stimuli that predict the arrival of food triggers the onset of the appetitive, or pleasure drive. This drive competes with the fear drive in the sensory-drive heterarchy. If the pleasure drive wins the competition, it releases incentive motivation, leading to an approach behavior in the presence of food. If the drive is not strong enough, no approaching behavior is released even in the presence of food.

Notice that the amount of time that the robot is not hungry depends on the amount of food it was able to take. In Fig. 19, at times near \( t=400 \) and \( t=740 \) the robot was able to eat enough food to last about 268 time units without being hungry again. However, a small food intake near \( t=1080 \) lead to a shorter satiation period, of about 200 time units.

We should point out that the proposed mechanism is not meant to be an accu-
Figure 19. The hunger homeostatic signal activation cycle. Positive values of the “Hunger” curve indicate that the hunger signal is active. Positive values of the “Eat” curve indicate food intake. Food is consumed only when the hunger signal is active and food is available. Hunger deactivates after food intake. The time elapsed before reactivation of the hunger signal depends on the size of the last food intake.

rate simulation of the typical hunger-satiety cycles seen in humans and animals. However, the point is to show how the model can be extended to include more realistic and useful situations in which drives and sensory stimuli interact in a more complex fashion. For instance, we envision replacing the lights with charging stations (which of course would have to be distinguishable by the sensors), in which case the pleasure drive would be directly correlated to the charge level of the battery, and after learning the robot would ignore charging stations until the battery level became sufficiently low.

Figure 20 shows the effect of the dynamical hunger signal on the robot’s approach behavior. The robot was placed in a small closed area. One of the walls had a 1” wide window, through which light was directed from the outside with a flashlight. A small closed area was chosen for this experiment to allow the robot to frequently encounter the source of light. Variations of this experiment can be conducted in larger environments with multiple or moving sources of light.

Initially the robot was hungry (Fig. 20(a)). It wandered until it found a source of light. At that moment the robot turned towards the light, approached it and stopped for several seconds, to indicate a food intake behavior. In Fig. 20(b), after enough food was taken, the robot continued its navigation. Notice that as a result of the approach behavior, the robot resumed its navigation head-
Figure 20. Effect of the pleasure drive activation cycle. (a) When hungry, the robot approaches the source of light until enough food has been consumed. (b) After food intake, the robot ignores the light.

ing directly towards the window. However, as the wall was detected, the robot quickly avoided the obstacle. The absence of a zig-zag movement similar to the one obtained in Fig. 16(b) indicates that no competition between pleasure and fear took place. Instead, only the obstacle avoidance behavior was observed. This occurred because after food intake, the hunger homeostatic signal became inactive due to satiation. When the hunger signal was absent, the light did not elicit pleasure, and the robot did not exhibit an approach behavior. As the robot kept moving, still satiated, it found and ignored the light once again.

With the use of the gated dipoles, we introduced some dynamics in the hunger homeostatic signal. This allowed the robot to approach sources of light only when it was hungry. When the robot was not hungry or no source of light was detected, the robot kept moving and exhibiting the obstacle avoidance behavior. If needed, more accurate models of animal feeding dynamics such as Staddon and Zanutto (1997) could also be integrated to our neural network.

8. Discussion

The neural network model we have proposed for avoidance and approach behaviors in real mobile robots has been inspired by Grossberg’s work on classical and operant conditioning. Without the need of supervision, the model is able to learn rapidly to avoid obstacles and to approach sources of light. Furthermore, since the model is largely independent of the nature and configuration of sensors,
it can be implemented on very different robotics platforms, as demonstrated by our experimental results.

The ability to work in the real world, with real sensors, on different robotics platforms, demonstrates the model’s success and robustness. In our opinion, this success is due primarily to our use of models of neural and behavioral aspects of animal learning. However, we are not the first to foresee the potential use of models of animal learning in robotics, nor is ours the only implementation of Grossberg’s models in real robots. Very impressive results have been reported by Baloch and Waxman (1991), using the robot MAVIN. They utilized a variant of Grossberg’s conditioning circuit as a part of the overall control scheme. The model they used is complex as it focuses on MAVIN’s visual navigation, attacking a variety of problems with specific solutions. The main difference is perhaps in our approach, since we are more interested in achieving rapid adaptive control that is independent of the platform on which the model is being used.

Closely related to our work is the implementation of the Schmajuk and DiCarlo model reported by Bühmeier and Manteuffel (1997). The basic network of Grossberg’s conditioning circuit is also used for an obstacle avoidance task in that work. However, this model differs from ours in several aspects. First, there is only one kind of learning, namely, the prediction of the UCS by the CS (classical conditioning). All the robot responses are prewired in the network, therefore all responses are regarded as reflexes in which no learning takes place. Second, as a consequence of this prewiring, knowledge of the robot’s geometry is required, since the reflexes make the right wheel turn back if collision is detected in the left front side of the robot, and so on. In contrast, our model requires neither knowledge of the robot configuration nor design of reflex behaviors. Instead, the obstacle avoidance and light approach behaviors arise due to learning.

From a different approach, Verschure and colleagues have proposed Distributed Adaptive Control (DAC) as an architecture that learns to generate approach and avoidance behaviors in response to appetitive and aversive stimuli (Verschure and Coolen, 1991; Verschure and Pfeifer, 1992; Pfeifer and Verschure, 1992; Verschure and Voegtlin, 1998). As with the model of Bühmeier and Manteuffel, built-in avoidance reflexes require knowledge about the sensor location on the robot’s body. Similarly, the model does not learn to generate new behaviors, but simply to generate built-in behaviors at the right time. Another major difference between DAC model and our model is that they specifically design the network in such a way that the obstacle avoidance behavior directly inhibits the approach behavior. Therefore, there is also a pre-built preference for the avoidance reflex over the approach reflex. In our case, no pre-wiring is required, since competition between the combination of drives and active cues determines which behavior will be released. Therefore, our network could eventually be expanded to include
more than two competing behaviors, all this without the need for further design considerations.

Finally, Nehmzow and colleagues have also used a neural network modulated by a sort of reinforcement signal to train robots to perform approach and avoidance behaviors (Nehmzow, 1995). In this work, learning in a simple two-layer feedforward associative network is modulated by the violation of what Nehmzow calls “instinct-rules,” essentially rules that describe sensory conditions under which certain instinctive behaviors should be released. A behavioral selection scheme chooses the appropriate behavior during learning in the following manner: inputs to the system generate a pattern of activity over the output layer, and the most active output layer node is used to generate a corresponding action. If this action leads to the violation of an instinct-rule, an internal mechanism selects the next most-active output node to see if the violation if removed. This selection cycle continues until a satisfactory action is taken. Here, as with Verschure’s DAC models, the network is essentially learning when to generate a pre-defined behavior in response to given sensory patterns.

Given the common points between our work and the models just summarized, a combined approach might be possible in order to extend further the learning capabilities of our neural network. For instance, we could combine basic reflexes of the type used by other authors in order to try to learn more complex behaviors, such as recognizing stimuli that cannot be represented by a single sensor.

9. Conclusions

We have described a model that learns to generate avoidance and approach behaviors for a wheeled mobile robot by using a form of “self-supervised” learning. The robot progressively learns to avoid obstacles without the need for external supervision, but simply through “punishment” signals produced by the collision of the robot during random exploratory motion. One of the main properties of the model is that it is necessary to know neither the robot’s geometry nor the configuration of the sensors on the robot’s surface. In our experiments with two different robotics platforms, i.e., Khepera and Pioneer 1, the same neural network learned to avoid obstacles, thanks to the model’s platform-independence. Moreover, learning in one environment generalized to any environment since it is based on the robot’s egocentric frame of reference. Experimental results with the Khepera robot showed that the neural network is also capable of learning to generate an approach behavior. Instead of “punishment” signals, “reward” signals were used to learn approach a source of light.

In Sec. 6 we extended the model of conditioning to account for multiple behaviors. Training for the avoidance and approach behaviors can be done simultane-
ously, even though these behaviors are quite opposite. With the addition of the hunger activation cycle we showed how drives and sensory stimuli can interact in complex situations. Although the coupled gated dipole described in Sec. 7 were not meant to be an accurate model of animal feeding dynamics, they allowed us to show how the robot can choose among different behaviors depending on the moment-by-moment combination of sensorial information and internal needs.

Using the extended model, we plan to utilize the visual system of the Pioneer 1 robot to learn to approach interesting objects found in the environment. This behavior would be equivalent to the light approaching behavior we have achieved using the Khepera robot. The use of visual information instead of infrared measurements would show further the platform-independence of our model.

We also want to combine our model with a neural network for low-level control developed by Gaudiano et al. (1996b), which allows the robot to learn in an unsupervised manner its inverse and forward kinematics using sensory feedback. The network constantly adapts to miscalibrations produced by wheel slippage, changes in the wheel sizes, and changes in the distance between the wheels. As is well known (Borenstein et al., 1996), odometry leads to accumulation of errors in the computation of the robot’s position. Although odometry sufficed for our target reaching experiments, complementary methods for navigation would be needed to maintain or improve positioning accuracy.

Finally, though versatile, purely reactive navigation is not enough for the target reaching task since the robot can get stuck in local minima paths. For this reason, we want to develop higher-level navigation schemes. Planning strategies combined with “frustration” when the performed plans fail should endow the robot with more powerful navigation skills.

References


Chang and Gaudiano


