Learning to Navigate from Limited Sensory Input: Experiments with the Khepera Microrobot

Roman Genov, Srinadh Madhavapeddi, Gert Cauwenberghs

Department of Electrical and Computer Engineering
The Johns Hopkins University, Baltimore, MD 21218-2686
E-mail: {roman, srinadh, gert}@bach.ece.jhu.edu

Abstract

The goal of this work is to augment reinforcement learning techniques for autonomous robot navigation with a state space encoding more representative of the actual state of the robot in its environment, than available from direct sensor readings. A second goal is to demonstrate the approach in a real-world setting, using the microrobot Khepera (K-Team, Lausanne, Switzerland). The choice of state representation is one of the most critical factors in the performance of reinforcement learning algorithms. The technique of inferring relative positional information indirectly from sensor readings, through unsupervised learning, is an important novel contribution of this work. As demonstrated in the robot experiments, the technique allows to optimally perform sensor fusion and avoids the need of more elaborate sensors conveying explicit information on position coordinates.

1. Introduction

In this paper we apply augmented reinforcement learning techniques to autonomous robot navigation with a state space encoding more representative of the actual state of the robot in its environment, than available from direct sensor readings. Section 2 briefly reviews reinforcement learning algorithms, specifically Q-learning. Section 3 addresses constraints imposed on a learning algorithm by real world environment. Section 4 describes our implementation of a hybrid neural network architecture that combines unsupervised learning in a Kohonen self-organizing map for state encoding of clustered sensor information, with Q-learning to map the coded discrete states onto control actions. Section 5 reviews the results of our work.

2. Q-learning

2.1. Reinforcement Learning

Reinforcement learning is a class of learning algorithms that map situations to actions as to maximize a reward signal. Learning is done interactively, by successive trial and error search. The agent moves from a state to a state as it performs actions. It receives externally supplied, delayed in time, discrete reinforcement signal which tells the agent how well or poor it is performing. The situation-to-action mapping, updated through the learning phase, constitutes the optimal policy for the agent at convergence. Credit assignment principles vary for different reinforcement learning algorithms (e.g. AHC [1], TD(λ) [2]). One of the most popular, Q-learning [3] , associates "quality" with a particular action in a given state.

This work was supported in part by NSF under Career Award MIP-9702346.

2.2. Q-learning

The goal of the learning agent in Q-learning is to maximize its expected return from each state X. The quantity $Q(X, a)$, which is the quality associated with action a in state X, represents the expected return of taking an action a and following a policy f thereafter, described by the value function:

$$V_f(X) = max_a Q(X,a), a \in A, \tag{1}$$

and the corresponding policy f that maximizes the expected returns:

$$f(X) = argmax_a Q(X,a)\ . \tag{2}$$

The general Q-learning update rule can be formulated as:

$$\begin{aligned} Q(X_t, a_t) &= \alpha(r_t + \gamma \cdot V_f(X_{t+1})) \\ &\quad +(1-\alpha)Q(X_t, a_t), \end{aligned} \tag{3}$$

where X_t represents the state at time t; a_t is the action taken in state X_t; α – the learning rate; r_t – reinforcement signal value at time t. At each iteration when the agent in state X_t moves to state X_{t+1} by taking action a_t, it updates $Q(X_t, a_t)$ a constant fraction towards the reinforcement received, r_t, and the value function, V_f, of the reached state X_{t+1}. For an appropriately decaying sequence of α and persistent activation of the entire state space, the Q-values converge with probability 1 to the expected discounted reinforcement and the learned policy is optimal [4]. However, Q-learning does not address many of the issues involved in learning in real world environment settings.

3. Learning in Real World Environment

Learning an optimal policy in real world settings, in tasks such as goal-oriented robot navigation, presents many new issues to be considered. Many reinforcement learning methods, including Q-learning, generally imply Markovian nature of the process. They assume a state encoding with transitional probabilities $P_a(X_{t+1}|X_t)$ that unambiguously describe the environment. However, in real world applications, such as robot navigation, sensors are often incomplete. In this case one has to learn a task in a partially observable environment where different physical locations may have identical sensory readings, and sensory values can not be used to represent states.

Another important issue in applying Q-learning to real world problems is computational complexity. The algorithm converges slowly or may fail to converge when the state and/or action space is large, such as in typical robot applications. Q-learning requires a sufficient degree of exploration (as oppose to exploitation) and in principle infinite time (to converge to the optimal policy), which makes generalization over a large input space difficult. In addition, real world environment is generally analog in nature, with various sources of noise (i.e. sensory-motor control errors). Variations in environmental conditions also have to be taken into account.

4. Goal-Oriented Robot Navigation

It has been shown that for simple tasks such as obstacle avoidance, raw sensory data is sufficient to learn the task [5]. For tasks of higher level of complexity, such as goal-oriented search while avoiding obstacles, a more efficient state encoding is needed.

We have implemented a hybrid neural network architecture that combines unsupervised learning in a Kohonen self-organizing map [6], for state encoding of clustered sensor information, with Q-learning to map the coded discrete states onto control actions. The purpose of the Kohonen map is to reduce the dimension of the state space, and extract relevant features corresponding to the topology of the physical environment. The adaptive clustering allows to generalize action-state pairs over the robot's state space and learn them more efficiently. The topology-preserving nature of the map ensures locality of the action-induced state transitions.

4.1. State Encoding

The Khepera microrobot (K-Team, Lausanne, Switzerland) has eight infra-red sensors, six of which are located in the front and two of which are located at the back of the robot. The sensors can be used both as proximity sensors for short range obstacle sensing as well as ambient light sensors . Two independent motors controlling the wheels allow the robot to move. Explicit 8-bit encoding of each of the 16 sensors data would account to $3.4 \cdot 10^{38}$ states in the Q-table. In order to reduce the size of the agent state-action space, states are represented by means of a self-organizing Kohonen map. It performs unsupervised clustering of the input vector space while preserving its local topology. The general algorithm was implemented as follows [6]:

```
1. Initialize the map near the centroid
2. Repeat:
    (a) Get input vector $I_t$
    (b) Determine the winner:
        $(i_{wta}, j_{wta}) = argmin(|T_t(i,j) - I_t|) \quad \forall i,j$
    (c) Update the template:
        for $i = i_{wta-1}, i_{wta}, i_{wta+1}$; $j = j_{wta-1}, j_{wta}, j_{wta+1}$;
        $T_{t+1}(i,j) = (1-\lambda_t) \cdot I_t(i,j) + \lambda_t \cdot S_t$
```

The robot environment consists of an arena surrounded with walls, and a light source in one corner defining the goal as shown in Figure 1. The action set included three actions of 100 ms duration each: go forward, turn left, and turn right. Under random policy, the sequence of raw sensory states, actions taken, and reinforcement signals was recorded over 45,000 iterations. The reinforcement was an external (human-supplied) discrete signal consisting of a punishment when the robot hit an object, and a reward when it reached the goal. The learning was performed in a batch mode on the data collected. We trained a 2-D Kohonen map with an 8×8 state space, S, encoding eight proximity and eight light sensor readings obtained from Khepera under random exploration.

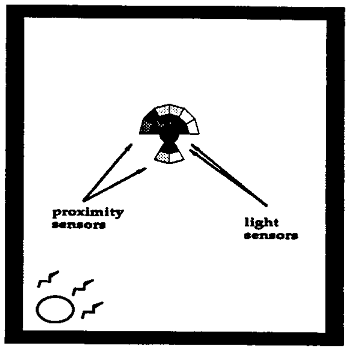

Figure 1: *Khepera in the arena, with eight proximity sensors (outer ring) and eight light sensors (inner ring).*

4.2. Learned State Map

The resulting 64-state Kohonen map is shown in Figure 2, where values of proximity and light sensors are represented according to the convention in Figure 1. The segments of the outer ring correspond to the proximity sensors; the segments of the inner ring correspond to the light sensors. Darker colored segments denote closer proximity to an obstacle or higher level of illumination respectively. The templates, shown on the grid, are clearly suggestive of typical configurations of the robot in the physical environment. For example, state (2,3) corresponds to the robot being in front of light, while state (2,0) indicating a corner (proximity sensors 2 and 4 have higher outputs than sensor 3). Neighboring nodes, on the other hand, retain many of the similar properties (such as neighboring location or small degree of rotation in physical space), while demonstrating a few different features. The sequence of states (7,0) through (3,0), for example, corresponds to the robot rotating while being close to a wall and far from the light source. Thus, the converged Kohonen map exhibits the expected generalization and neighborhood preserving properties.

To verify convergence of the learned Kohonen map two techniques were used. One way is to check that all the nodes on the grid are equally accessed. We keep track of the activity levels of the nodes during the Kohonen iteration:

$$A_{t+1}(i,j) = (1-\mu) \cdot A_t(i,j) + \mu \cdot \delta_{i_{wta}, j_{wta}}, \forall i,j, \qquad (4)$$

where $A(i,j)$ is activity level of node (i,j); $\delta_{i,j}$ is 1 if the node (i,j) is selected, 0 – otherwise; μ is the learning rate. Distribution of the activation levels over all nodes is checked for uniformity. We also examined histograms of the change in i and j index values on the Kohonen grid (Δi and Δj) values, at the beginning of the iteration process over 1000 iterations, and again over 1000 steps

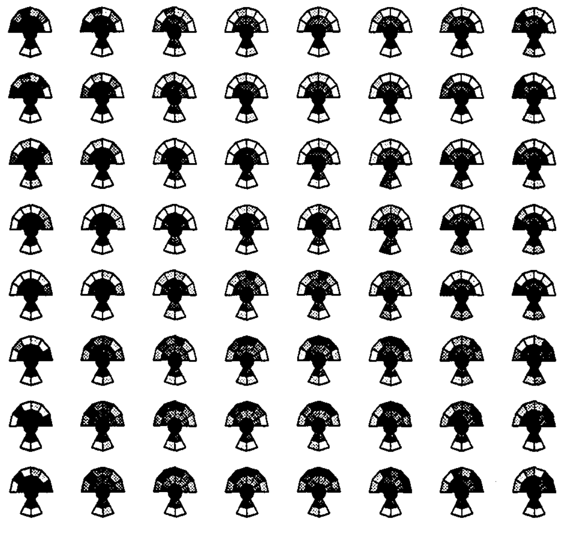

Figure 2: *The converged Kohonen map of sensory data. The templates, shown on the grid, include proximity sensors (the segments of the outer ring) and light sensors (the segments of the inner ring). Darker colored segments denote closer proximity to an obstacle or higher level of illumination for two types of sensors respectively.*

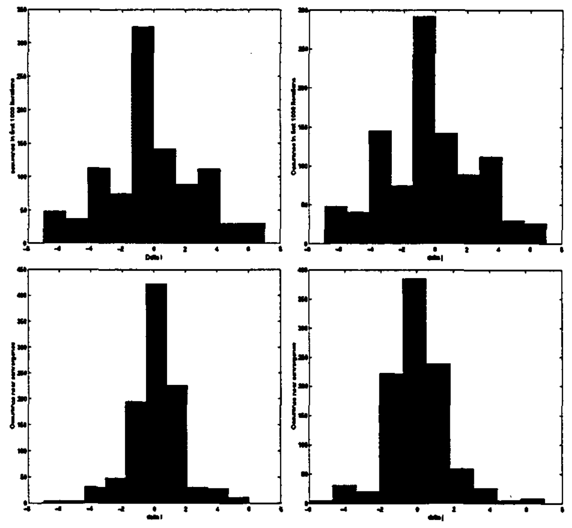

Figure 3: *The histograms of node distances for consecutive state transitions on the Kohonen map at different stages of the learning process: in horizontal (top) and vertical (bottom) directions on the grid; at the beginning of learning (left) and at convergence (right).*

at convergence. The histograms, shown in Figure 4.2, suggest that at convergence, the states that the agent transitions to are indeed neighboring.

4.3. Q-learning Algorithm Implementation

In order to implement batch-mode Q-learning on the Kohonen map, we need to obtain the learned statistics of state transitions from the data sequence recorded from the robot. These statistics, for a given present state and action, were estimated by averaging over all data entries:

$$X'(X_t, a_t) = (1-\nu)\cdot X'(X_t, a_t) + \nu \cdot X_{t+1}, \qquad (5)$$

where $X'(X_t, a_t)$ is the estimated position on the grid, with coordinates i', j', that the present state $X(i,j)$ leads to, X_{t+1}, obtained as a moving average over a large number of samples. Thus, the average next state for a given state-action pair yields a continuous valued vector, off the grid, on the Kohonen map. Without using interpolation or other continuous-space techniques (e.g. GTM [7]), discrete nature of the Kohonen map allows us to construct only a coarse state transition map.

To perform Q-learning, we implemented an algorithm based on value iteration with sample backup [8]. Sample backup is a model-free method that samples the next state from a distribution over all possible successor states. For all of the sates X, we estimate state transition probabilities from the state to a few (i.e. four) Kohonen map nodes (denoted as set $N \in S$) that are possible successor states and are the closest neighbors of the average next state calculated in (5) (see Figure 4):

$$P_a(X_i|X) = \frac{w(X'(X,a), X_i)}{\sum_i w(X'(X,a), X_i)}, \qquad (6)$$

where $w(X'(X,a), X_i)$ is a radial basis projection of the average next state $X'(X,a)$ onto one of the neighboring states X_i; and i is a permissible index among all states in the closest neighborhood of the average next state as defined above (set N). The iteration is performed as follows:

$$\begin{aligned} Q(X,a) &= \alpha\big(\hat{r}(X) + \gamma \sum_i P_a(X_i|X)\cdot max_{a'} Q(X_i, a')\big) \\ &\quad +(1-\alpha)\cdot Q(X,a), \end{aligned} \qquad (7)$$

where reinforcement is sampled over set N with the estimated average

$$\hat{r}(X) = (1-\chi)\hat{r}(X) + \chi r_t\,. \qquad (8)$$

The resulting plot of maximum Q-values over all possible actions is presented in Figure 5.

5. Navigation Results

The Kohonen map at convergence indicates clustering of sensor values into ordered classes that physically correspond to relative position of the robot in the environment, such as distance and angle to walls, corners and light source. Regression of state transitions under control actions confirm "continuity" of the topology-preserving map, and value iteration of the Q map leads to a control policy that steers the robot towards the goal while avoiding obstacles. Finally, we tested the robot autonomously navigating in its environment using the learned Kohonen and Q-optimal policy maps. The converged policy map and experimental on-line navigation trajectories shown in Figure 6 confirm that the robot has learned the navigation task: starting at different initial states, it navigates to the goal state (2,3) avoiding obstacles.

6. Conclusions

A combination of unsupervised and reinforcement learning techniques has been proposed as an alternative to explicit position state encoding, for autonomous agents interacting with an unknown environment through incomplete sensors of which the characteristics are equally unknown. Experimental results on the Khepera microrobot confirm successful learning of a simple obstacle avoidance and goal-directed search task, using only limited-range proximity and ambient light sensors.

7. References

[1] A. Barto, R. Sutton, and C. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 13(5), pp. 834-846, 1983.

[2] P. Dayan, "The convergence of TD(λ) for general," *Machine Learning*, vol. 8(3), pp. 341-362, 1992.

[3] C. Watkins, P. Dayan, "Q Learning" *Machine Learning*, vol. 8(3), pp. 279-292, 1992.

[4] T. Jaakkola, M. Jordan, and S. Singh, "On the convergence of stochastic iterative dynamic programming algorithms,". *Neural Computation*, vol. 6(6), November 1994.

[5] C. Touzet, "Neural Reinforcement Learning for Behavior Synthesis," *Robotics and Autonomous Systems*, vol. 22, pp. 251-281, 1997

[6] T. Kohonen, "Self Organization and Associative Memory," Springer-Verlag, Berlin, 1984.

[7] C. Bishop, M. Svensen, and C. Williams, "GTM: A Principled Alternative to the Self-Organizing Map," In M. Jordan, et al, editors, *Advances in Neural Information Processing Systems*, vol. 9, pp. 354-360, The MIT Press, Cambridge, MA, 1998.

[8] S. Singh, "Learning to Solve Markovian Decision Processes," PhD thesis, Department of Computer Science, University of Massachusetts, 1993; or CMPSCI Technical Report 93-77.

[9] L. Kaebling, M. Littman, and A. Moore, "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, vol. 4, pp.237-285, 1996.

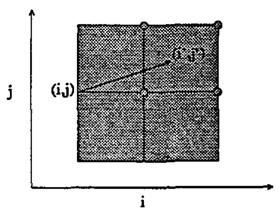

Figure 4: *An illustration of the method of sampling the next state from a distribution over all possible successor states. Experimental state transition probabilities are evaluated over four Kohonen map nodes (circled) that are the closest neighbors of the average next state (i', j')*

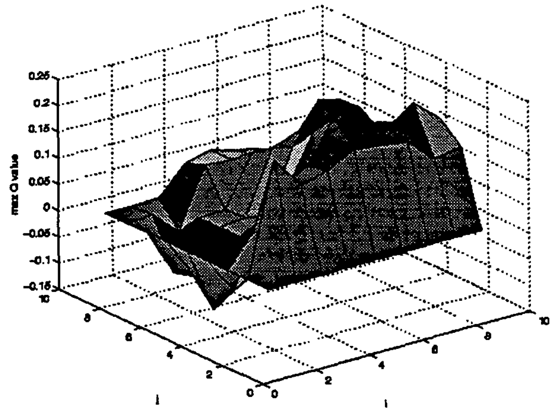

Figure 5: *The maximum Q-values over all possible actions.*

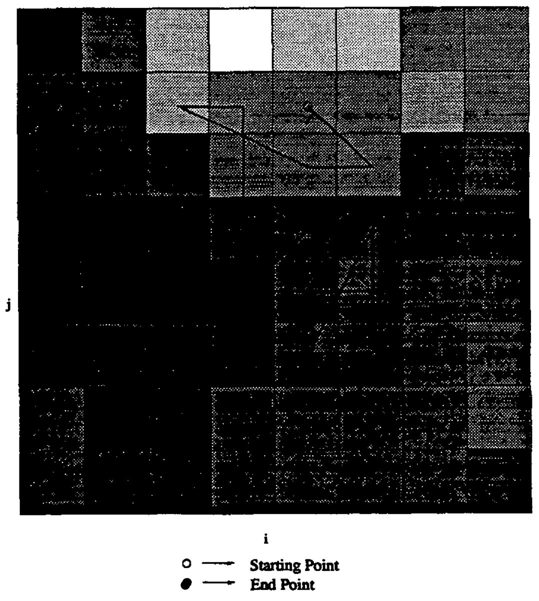

Figure 6: *The optimal-Q map and robot trajectories obtained online, by navigating the robot according to the learned policy.*