Reinforcement Learning for Landmark-based Robot Navigation *

Dídac Busquets, Ramon López de Màntaras, Carles Sierra
Artificial Intelligence Research Institute (CSIC)
Campus UAB, 08193 Bellaterra, Spain
{didac,mantaras,sierra}@iiia.csic.es

Thomas. G. Dietterich
Oregon State University
Corvallis, OR, 97331 USA
tgd@cs.orst.edu

1. INTRODUCTION

In landmark-based navigation, robots start in an unknown location and must navigate to a desired target using visually-acquired landmarks. In the scenario that we are studying the target is visible from the robot's initial location, but it may subsequently be occluded by intervening objects. The challenge for the robot is to acquire enough information about the environment so that it can, even in that case move, from the starting location to the target position.

In this paper, we build upon our previously described multiagent system for outdoor landmark-based navigation [2]. It is composed of three systems: the Pilot, responsible for all motions of the robot, the Vision system, responsible for identifying and tracking landmarks and for detecting obstacles, and the Navigation system, responsible for choosing high-level robot motions.

These three systems must cooperate to achieve the overall task of reaching the target. For instance, the Pilot needs the Vision system to identify obstacles and the Navigation system to select a path to the goal. The systems are also competing, for instance, the Pilot and the Navigation system both compete for the Vision system. The Pilot needs it for obstacle avoidance, while the Navigation system needs it for landmark detection and tracking.

To manage this cooperation and competition, in [2] we had chosen a bidding mechanism. Each system generates bids for the services offered by the Pilot and Vision systems. The service actually executed by each system depends on the winning bid at each point in time. In [2] we proposed bidding functions to obtain good performance from the combined system. In this paper we use Reinforcement Learning (RL) [5] to tune the parameters of those functions.

The Navigation system is also implemented as a multiagent system composed of six agents with the following goals: keep the target located with maximum precision and reach it (*Target Tracker*), keep the risk of losing the target low (*Risk Manager*), recover from blocked situations (*Rescuer*), keep the error in the distance to landmarks low (*Distance Estimator*), and keep the information on the map consistent and up-to-date (*Map Manager*). The Navigation system also employs a bidding mechanism to coordinate the agents. Each agent bids for the action it wants the robot to perform. An agent called *Communicator* determines the winning action.

For map representation and wayfinding, we employ an extension of Prescott's beta-coefficient model [4]. In this model the location of a landmark is invariantly encoded based on the relative locations of three other landmarks. As the robot explores the environment, it stores the relationships among the landmarks it sees. Prescott's model assumes that the robot is able to measure the exact location of the landmarks. We have extended this model to deal with imprecise locations by means of fuzzy arithmetic [1].

RL is most appropriate in cases where there is a complex, quantitative tradeoff between behaviours. In such cases, manual tuning is difficult. Instead, the quantitative criterion of maximizing expected reward, which is the goal of RL, permits us to represent the tradeoff nicely. Within the Navigation system, such a tradeoff exists between the *Target Tracker*, the *Risk Manager*, and the *Distance Estimator*. For instance, the *Target Tracker* wants to know the exact heading and distance to the target at all times, while the *Risk Manager* wants to ensure that the robot is surrounded by a rich network of landmarks so that the robot does not get lost. These agents' goals compete for the control of the camera.

In this paper, we propose to replace the *Target Tracker*, the *Risk Manager*, and the *Distance Estimator* with a new *Learning Agent* (LA) whose bidding function (known in RL as the value function) will be tuned by RL. We will formulate the reward function for this agent so that it is rewarded for reaching the current target location while minimizing the use of the camera.

2. THE TASK TO BE LEARNED

The task confronting the *LA* is to choose actions (for both motion and vision) to reach the current target location while minimizing the use of the camera. If the robot becomes blocked, the *Rescuer* will choose a diverting target, and then the *LA* will take control and choose actions to reach that new target. Once the diverting target is reached, the *Rescuer* may be able to set the current target to be the initial one, and then the *LA* will resume the moving to that target.

3. THE RL ALGORITHM

There are two general styles of RL algorithms: Model-based and Model-free. Model-based algorithms learn a transition model for the environment, $P(s'|s,a)$, where s is the state of the environment at time t, a is an action to be executed, and s' is the resulting state of the environment at time $t+1$. These algorithms also learn a reward model $R(s,a,s')$ which gives the expected one-step reward

* Research supported by the US-Spain Joint Commission for Scientific and Technological Cooperation and Plan Nacional Project DPI 2000-1352-C02-02. Dídac Busquets holds the CIRIT doctoral scholarship 2000FI-00191.
Long version: http://www.iiia.csic.es/∼didac/AR/AutonomousRobots.html

of performing action a in state s and making a transition to state s'.

For robot learning model-free methods are impractical, because they require many more interactions with the environment to obtain good results. Hence, for our experiments, we have chosen the model-based algorithm known as Prioritized Sweeping [3].

We represent both the transition model $P(s'|s,a)$ and the reward model $R(s,a,s')$ by three-dimensional matrices with one cell for each combination of s, s', and a. This technique only works if the state and action spaces are small. Hence, the most challenging aspect of applying RL is the proper design of the state representation.

3.1 State Representation

To explain our state representation, we begin by defining a set of belief state variables and how they are discretized to provide a small set of features each taking on a small set of values so that $P(s'|s,a)$ and $R(s,a,s')$ can be represented with small tables.

The headings to all objects are divided into six sectors of 60 degrees. For each sector, we represent information about the landmarks believed to be in that sector and their imprecision. Given these sectors, the following state variables can be defined: distance to target, $D(t)$ and its imprecision, $I_d(t)$; heading to target, $H(t)$ and its imprecision, $I_h(t)$; the landmarks in each sector, $L(s) = \{l_1, ..., l_{n_s}\}$; number of landmarks in each sector, $N(s) = \min(4, |L(s)|)$; and the average imprecision of landmarks in each sector, $\overline{I}(s) = \frac{1}{N(s)} \sum_{l \in Best(4,L(s))} I(l)$.

The distance $D(l)$ to a landmark ($D(t)$ to the target) is a fuzzy number in the range $[0, \infty]$. The heading to a landmark $H(l)$ ($H(t)$ to the target) is a fuzzy number with range $[0, 2\pi]$. For each of these, the imprecision ($I_d(l)$ for distance, $I_h(l)$ for heading) is defined by taking the interval corresponding to the 70% α-cut of the fuzzy number. The imprecision of a landmark is computed by combining the imprecision in the heading and in the distance. We summarize the agent's knowledge of the landmarks in each sector by averaging the imprecision of the four most-precisely-known ones.

From these state variables we define the following features: *target distance*, ($D(t)$), discretized to 5 intervals; *target location imprecision*, ($I(t)$), discretized to 7 intervals; *landmark count*, $\overline{C} = \frac{1}{6}\sum_{s=0}^{5} N(s)$, discretized to 4 intervals; and *landmark imprecision*, $\overline{I} = \frac{1}{6}\sum_{s=0}^{5} \overline{I}(s)$, discretized to 7 intervals.

3.2 Actions

The actions we have designed for the *LA* are: *Move Blind (MB)*, move in the direction in which the target is believed to lie, not using the vision system; *Move and Look for Landmarks (MLL)*, move toward the target, point the camera in the sector containing the fewest number of known landmarks, and look for new landmarks in this sector; *Move Orthogonally to Target (MOT)*, move orthogonally to the direction of the target, point the camera at the target and attempt to improve the precision of its heading and distance; *Move and Verify Landmarks (MVL)*, move toward the target, point the camera to the sector with the maximum imprecision $\overline{I}$, and attempt to re-acquire known landmarks and measure their heading and distance more accurately; and *Move and Verify Target (MVT)*, move toward the target, point the camera at the target and attempt to re-acquire it and measure its heading and distance more accurately.

The rewards are negative, as they model costs. The actions have the following costs: MB is the cheapest action, and has a reward of -1; MVL and MVT, -5; MOT, -6; MLL has a reward of -10, as it must do extensive image processing. The LA receives a reward of 0 when it reaches the target location.

4. EXPERIMENTATION AND RESULTS

We use the Webots simulator (www.cyberbotics.com) to perform our experiments. The simulated environment contains a set of landmarks, one of them being the target. There is a wall that surrounds the region in which the robot is navigating. The landmarks are the only objects in the environment. In each trial, the robot starts at a random location, and it has to reach the target. Each trial terminates if: (a) the robot reaches the target, (b) the target is not reached within a given time limit, or (c) the robot is blocked. We compare the performance of the LA with a hand-coded policy (HC).

Table 1: Comparison of *LA* and HC after 2000 training trials.

	Reward per trial	Actions per trial	MB	MOT	MVT	MVL	MLL
HC	–858	153.33	4.94	18.59	0.52	121.96	7.32
LA	–336	49.95	11.41	6.52	5.61	4.97	21.43

The *LA* was trained for 2000 simulated trials. At regular intervals, the learned value function was tested by placing the robot in 100 randomly-chosen starting locations, running one trial from each location, and measuring the total reward, the total number of actions, and whether the robot succeeded in reaching the target. The same set of starting locations was employed in each testing period. The HC was also evaluated on these starting locations. After only 100 trials, the *LA* is already out-performing the HC. After 2000 trials, the *LA* succeeds in reaching the target in 84 of the trials, compared to only 24 for the HC.

From the results in Table 1, we see that while the HC receives an average of -858 units of reward, the LA receives -336 units, which is a huge improvement. In addition, the *LA* on the average is three times faster than HC and it never terminates because of reaching the time limit. We also see that the *LA* has learned to perform fewer MOT and MVL actions and more MB, MVT, and MLL actions. One of the goals of applying RL was to find a policy that freed the camera for use by the low-level obstacle avoidance routines, and this is exactly what has happened (11.4 MB actions for LA, and 4.9 for HC). On the other hand, we were surprised to see that the *LA* chooses to execute the most expensive action, MLL, so often (21.4 times per trial, compared to only 7.3 for the HC). Evidentally, it has found that a mix of MLL and MB gives better reward than the combination of MVL and MOT that is produced by the HC. The *LA* spends much more time looking for new landmarks and much less time verifying the direction and distance to known landmarks.

5. FUTURE WORK

The next step in our work is to integrate the *LA* into the overall multi-agent system and compare the performance of the *LA* with the performance of the hand-coded bidding functions described in [2]. Additional future work will also test the robustness of these results to changes in the set of features and the set of actions.

6. REFERENCES

[1] G. Bojadziev and M. Bojadziev. *Fuzzy sets, fuzzy logic, applications*, volume 5 of *Advances in Fuzzy Systems*. World Scientific, 1995.

[2] R. López de Màntaras C. Sierra and D. Busquets. Multiagent bidding mechanisms for robot qualitative navigation. In *Intelligent Agents VII. LNAI (ATAL-2000)*, pages 198–212. Springer, Verlag, 2001.

[3] Andrew W. Moore and Christopher G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13:103, 1993.

[4] T.J. Prescott. Spatial representation for navigation in animats. *Adaptive Behavior*, 4(2):85–125, 1996.

[5] Rich Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, 1998.