A Reinforcement Learning Approach for a Goal-reaching Behavior

T. D'Orazio and **G. Cicirelli** and **G. Attolico**
Istituto Elaborazione Segnali ed Immagini - C.N.R.
Via Amendola, 166/5 - 70126 Bari (Italy)

C. Distante
Computer Science Department
University of Massachusetts, Amherst MA

Abstract

Developing elementary behavior is the starting point for the realization of complex systems. In this paper we will describe a learning algorithm that realizes a simple goal-reaching behavior for an autonomous vehicle when a-priori knowledge of the environment is not provided. The state of the system is based on information received by a visual sensor. A Q-learning algorithm associates the optimal action to each state, developing the optimal state-action rules (*optimal policy*). A few training trials are sufficient, in simulation, to learn the optimal policy since during the test trials the set of actions is initially limited. The state and action sets are then enlarged, introducing fuzzy variables with their membership functions to the extent of tackling errors in state estimation due to the noise in the vision measurements. Experimental results, both in simulated and real environment, are shown.

Introduction

Studies regarding intelligent robots have received a great attention in the scientific community in the last years both for the biological aspects involved, and for the robots' physical structure required. The definition of intelligence can assume different meanings in the robotic field. The one that seems more promising has been used by Arkin in (Arkin 1998): "An intelligent robot is a machine able to extract information from its environment and use knowledge about its world to move safely in a meaningful and purposive manner". Behavior-based systems (Brooks 1991) have shown that some kind of intelligence is able to emerge by reacting to what it can be perceived in the environment. These systems use behaviors as a way of decomposing the control policy needed to accomplish a task and are very useful in making robots adapt itself to the dynamics of the real world environment. However, the behaviors in these systems are usually organized in a fixed and pre-defined manner, giving less flexibility for managing a high number of goals and situations. For this reason, the need of robots that learn robust behaviors in a real environment has increased. A number of architecture has been devised in literature to take care of different behaviors: hierarchies of behaviors, concurrent parallel behaviors, or layered structured hierarchies. Central control modules for behavior selection or hard-wired arbitration networks have been used according to the complexity of the task involved. Whatever strategy is chosen the primary advantage of the task decomposition is that each behavior has to solve the perception, modeling, and planning problems relevant to the particular task for which it is designed. A second advantage for this kind of architecture is that they can be developed incrementally adding new behaviors as new capabilities are needed. Thus developing elementary behaviors is the starting point for the realizations of complex systems (Mahadevan & Connell 1992; Caironi & Dorigo 1997). One of the problems with behavior-based robots is that the component modules have to be laboriously programmed by a human designer. If new behaviors could be learned, the designer would be free from understanding the interactions between a particular robot and the environment.

This work regards the learning of an elementary behavior that can be included in a controller architecture that realizes a complex task. The robot has no task knowledge to begin with, but it has to learn from the interactions of the environment. Reinforcement learning techniques seem suitable for addressing this problem: the agent chooses an action based on its current and past sensor values and it maximizes over time a reward function measuring the agent's performance. The use of visual information has been limited in literature because of the difficulties and the cost of processing visual data (Srinivasan & Venkatesh 1997; Nayar & Poggio 1996). Actually sonar sensor, odometry and proximity sensors have been used to solve elementary behavior that however are limited only to local tasks. Visual sensors, instead, can be more useful since they are able to detect distant goals and permit the acquisition of suitable behaviors for more global and goal-directed tasks (Asada *et al.* 1996; Mahadevan, Theochaous, & Kaleeli 1998).

A goal-reaching behavior has been realized using qualitative information of the vehicle position provided by a vision system. The image captured by the camera placed on the vehicle, gives information of the system state according to the vehicle position with respect to the goal: (Near, Medium, Far) and (Front, Left-side, Right-Side). These states are defined as regions in the environment. During the learning phase the proper action to reach the goal is associated to each state. We

use a Q-learning algorithm, a form of model-free reinforcement learning based on a stochastic dynamic programming, to find the optimal state-action mapping. The learning phase has been executed in simulation, afterwards a testing phase has been carried out in both simulation and real experiments. In the testing phase the knowledge learned is checked by forcing the vehicle to choose always the optimal actions in each state while it performs some paths from different starting positions to the goal. The paths obtained have abrupt changes of directions because of the limited action space and the state classification used. Besides in the real environment the noise in image data does not allow accurate estimates of the actual states of the vehicle. These considerations brought us to the introduction of transition areas between the regions of the environment and to a new definition of the state of the robot. Fuzzy variables with the relative membership functions have been introduced to compute both the states and the corresponding actions that have to be performed by the robot. This determines a gradual change of the robot's state through adjacent regions and an increase of the number of actions. As a consequence during the navigation the robot performs smooth paths.

The paper is organized as follows. The following section defines the elementary behavior giving a full description of the state and action sets used in the learning algorithm. Successively the experiments, realized with both the simulated and the real robot, are shown. Results obtained with and without the fuzzy regions extension are also compared. Finally some conclusions are drawn.

The Elementary Behavior

In this paper we present a method of vision-based reinforcement learning for an elementary goal-reaching behavior in a closed and free environment. The robot does not need any information about the environment. The image captured from the camera mounted on the robot is the only source of input for the algorithm.

Once the goal state has been fixed, an analysis of the input image allows the detection of the current system state s_t relative to the goal state. An action a_t is selected by using an action selection function. The robot performs the action a_t in the state s_t and it enters in the new state s_{t+1}. The environment returns a reward value r_t which judges the just executed action. Using this information the learning algorithm updates the control policy which maximizes the expected discounted sum of future reinforcements. The algorithm used to find this optimal policy is the well known Q-learning algorithm (Watkins & Dayan 1992).

The State Set

This section defines the state of the system and how it is obtained. Since the aim of this work is to test the learning algorithm when only qualitative information is available, we used a simple quantitative information obtained from a visual system. The trick used is described as follows: the environment has been split in regions, as shown in Fig. 1, representing the state of the vehicle. In particular the state s_t at time t is defined as the couple (i, j) where $i \in$ {LEFT-SIDE,FRONT,RIGHT-SIDE}, $j \in$ {NEAR,MEDIUM,FAR}. The state (FRONT,NEAR) has been fixed as the goal state. The visual system localizes the robot in one of the regions of the environment using a simple self-location method. This method, described in detail in (E.Stella & A.Distante 1995), estimates the robot position in an absolute coordinate system $(X - Y)$ using three landmarks of the environment. In order to speed up the state detection phase, artificial landmarks (infrared led) have been placed in the environment since their identification in the image is simple and fast. A landmark tracking module for keeping the landmarks centered in the image has also been implemented.

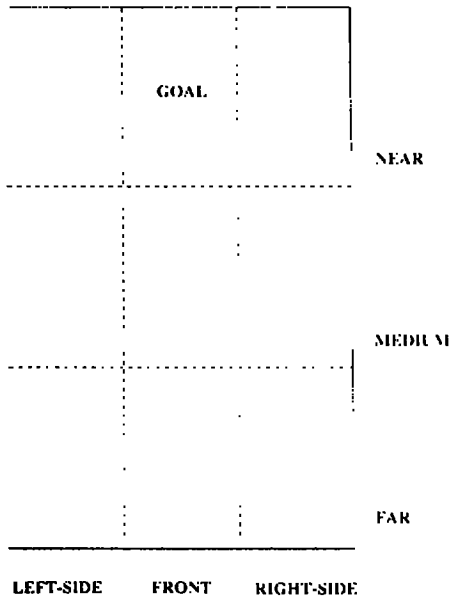

Figure 1: State Space

The Action Set and Q-learning

This section describes the core of the learning process: the mapping from states to actions required for solving the considered task. The action space is discrete and contains eight actions corresponding to the absolute orientations that the agent can take in each state: $\mathcal{A} = \{0, 45, 90, ..., 315\}$ $(degrees)$. Performing one action means that the vehicle rotates until it is oriented towards the selected direction and translates until the current state changes.

The purpose of the learning process is to learn for each state s_t the best action a_t. The learning algorithm used to find this mapping is the Q-learning algorithm (Watkins & Dayan 1992). This is based on the estimation of the real-valued function $Q(s_t, a_t)$ that gives the expected discounted sum of future reinforcement for performing action a_t in state s_t. At each time step the

value of the estimate $Q(s_t, a_t)$ is updated as follows:

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha(r_t + \gamma \max_{b \in \mathcal{A}} Q(s_{t+1}, b)) \quad (1)$$

where s_{t+1} is the next state after taking action a_t in state s_t, α is the learning rate ($0 \leq \alpha < 1$), γ is a discount factor ($0 \leq \gamma < 1$), r_t the reinforcement value. This value is an evaluation of the current system performance which can be successful or not in relation to the desired behavior. In our application the system receives $r_t = 1$ if it reaches the goal-state, $r_t = -1$ if it bumps into a wall and $r_t = 0$ in all the other state transitions. This is a sort of *delayed reinforcement* used in those applications where it is difficult to evaluate the performance of the system at each step. In our case reinforcements are available only when the performing system achieves given states, *i.e.* either when it enters the goal-state (reward) or when it bumps into a wall (penalty).

The action selection is carried out in the following way. Whenever the system encounters a new state a random selection with uniform distribution is applied to the set of the eight possible actions. A stochastic selection, with the Boltzmann distribution, is applied when the system is in a state already visited. Indeed the probability of choosing the action a_t in the state s_t depends on the Q-value $Q(s_t, a_t)$ as follows:

$$p(a = a_t | s = s_t) = \frac{e^{Q(s,a)/T}}{\sum_a e^{Q(s,a)/T}} \quad (2)$$

where T is the temperature parameter. The Boltzmann distribution permits a smooth transition from pure exploration to exploitation as the experiment goes on. This is done tuning opportunely the temperature parameter from higher to lower value.

The Experiment

A number of experiments have been performed to test the visual goal-reaching behavior described in the previous sections. A simulation phase and a testing phase have been considered. In the simulation phase the state-action mapping is completely learned, while in the testing phase the optimality of the acquired behavior is evaluated in both the simulated and real environment.

The robot

A Nomad200 platform, that integrates a mobile robot system with several sensory modules, has been used in our experiments. Only two sensory systems are considered: the 20 tactile sensors mounted around the body of the vehicle and the video camera mounted on a pan/tilt head on the top of the vehicle. The tactile sensors can detect collisions with the walls in the environment, while the camera continuously captures images of the environment. The pan/tilt head can rotate 174 degrees horizontally and 70 degrees vertically.

The Simulation Phase

The described learning procedure has been initially experimented using the Nomad200 simulator provided by the Nomadic Inc. The simulated environment has been delimited according to the real environment making the successive transfer of the learned policy from the simulated to the real test-bed possible.

At each step the current state s is computed by using the information received by the robot position, then an action a is selected and performed bringing the robot in a new state. The $Q(s, a)$ Q-values are updated according to the (1) formula. In order to obtain a dense matrix of the Q-values a number of trials has been repeated to guarantee the visit of all the states. A trial is the execution of a number of paths from different starting positions fixed in the environment. Each path can terminate with a success if the goal-state is achieved or with a failure if a collision occurs. The starting positions considered are 22 different points at the borders of the environment. The environment is 140*inches* wide and 200*inches* long. During the simulation, after 20 trials, we have verified that the Q-values reached their stable values. Fig. 2 shows the optimal actions (*i.e.* with the highest Q-value), obtained after this learning phase: the dotted lines delimits the states of the system in the environment, in each state the orientation that has to be taken by the robot is depicted.

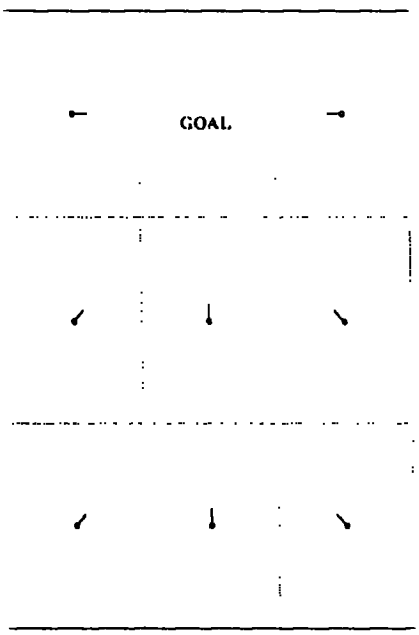

Figure 2: Optimal actions learned after the learning phase in simulation.

The Testing Phase

During the testing phase the learning algorithm is switched off and a number of paths are performed by the agent in both simulated and real environment. Fig. 3 shows some simulated paths obtained after the completion of the learning phase.

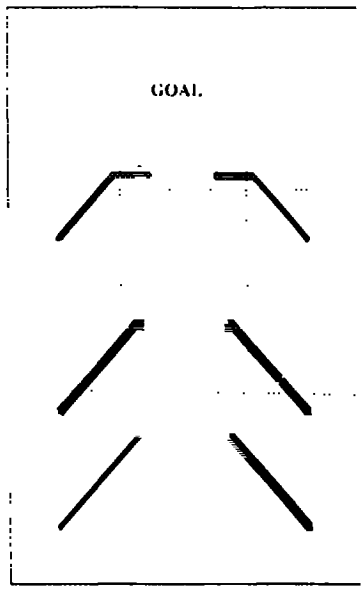

Figure 3: Simulated paths

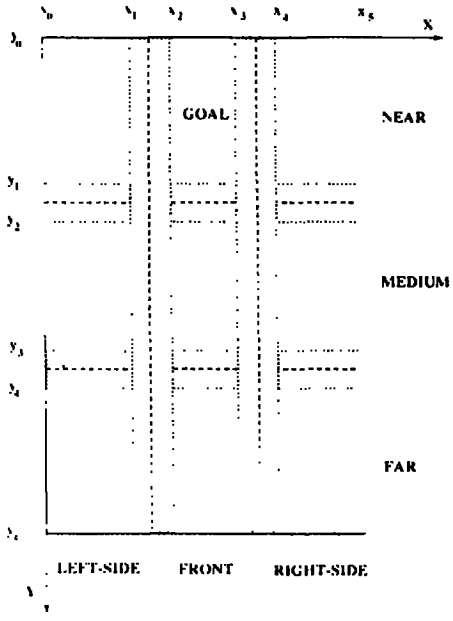

Figure 4: State Space with transition areas

In order to examine the optimality of the trajectories found by the robot, the paths are performed by forcing the choice of the optimal actions. The abrupt changes of the actions taken by the robot in Fig. 3 are due to both the limited action space and the splitting of the environment into regions. In a real environment the inaccurate estimation of the actual state, due to the peculiarity of visual information, develops the same type of discontinuities. This happens above all at the borders of the regions where small errors in the position estimation may result in a wrong evaluation of the actual state. Therefore we decided to define transition areas between the regions (see fig.4) adopting a fuzzy model for the state definition. Fuzzy sets are a powerful tool for representing sets with fuzzy boundaries. We have considered two fuzzy variables: X ranging in the set $\{LEFT-SIDE, FRONT, RIGHT-SIDE\}$ and Y ranging in the set $\{NEAR, MEDIUM, FAR\}$ where the six labels denote the fuzzy sets shown in Fig. 5.

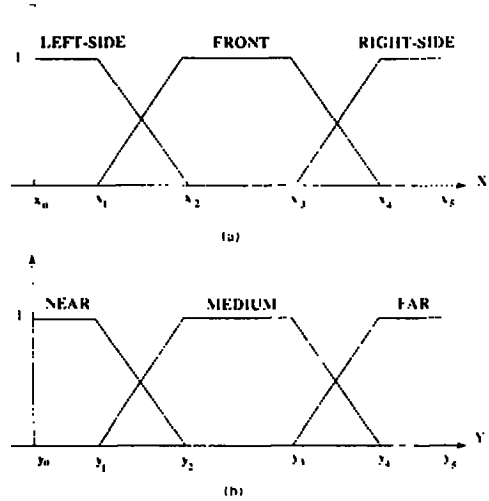

Figure 5: Respectively membership functions of the fuzzy variables X (a) and Y (b).

In the figure the ranges of the continuous values for each fuzzy set are also displayed, they are delimited by the symbols x_k and y_k $(k = 0 \ldots 5)$. These values have been fixed heuristically according to the error in the robot position estimation. Each continuous value can belong to a fuzzy set with a degree of membership μ in the range $[0, 1]$. The membership functions of the X and Y fuzzy variables are shown respectively in Fig. 5(a) and (b). With the introduction of membership functions a given value x is no more considered as completely $LEFT-SIDE$, $FRONT$ or $RIGHT-SIDE$ but if the value x is in the range $[x_0, x_1]$ it is considered as completely $LEFT-SIDE$ $(\mu(x) = 1)$, whereas if x is in the range $[x_1, x_2]$ it is considered as $LEFT-SIDE$ with a given μ_x value and as $FRONT$ with a $1 - \mu_x$ value and so on. The interesting property of fuzzy sets admits the partial overlapping of their membership functions with a gradual change in the classification of contiguous values as belonging to adjacent regions. The state of the system is now defined by the quadruple (i, j, μ_i, μ_j), where i and j are the same as described in the section "The State Set" and represent two fuzzy sets, whereas μ_i and μ_j are the values of membership of the x y position values to the i j fuzzy sets. As a result an action a^* is now computed according to the formula:

$$a^* = \frac{(\mu_i * a_i^{max} + \mu_{i+1} * a_{i+1}^{max} + \mu_j * a_j^{max} + \mu_{j+1} * a_{j+1}^{max})}{\mu_i + \mu_{i+1} + \mu_j + \mu_{j+1}} \quad (3)$$

where a_i^{max} and a_{i+1}^{max} are the optimal actions shown in Fig. 2 of the two adjacent regions i and $i+1$ in the ordered set $\{LEFT-SIDE, FRONT, RIGHT-SIDE\}$, whereas a_j^{max} and a_{j+1}^{max} are the same but with j in the set $\{NEAR, MEDIUM, FAR\}$. As a result the set of actions becomes continuous. The new definition of the system state and the new action space allow the robot to perform smooth paths as we can see in figures 6 and 7 in both the simulated and real test-beds.

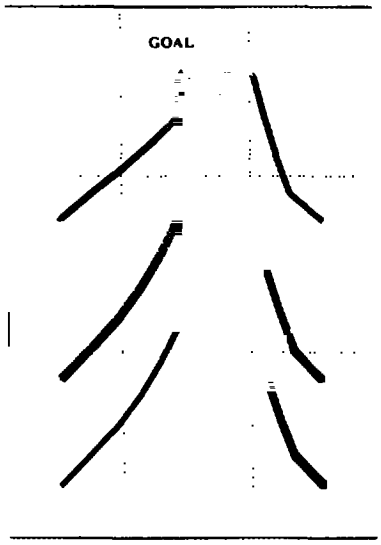

Figure 6: Simulated paths with the fuzzy regions extension

Conclusions

This work claims that it is possible to learn good action strategies, from reinforcement, for vision-based systems moving in a real and noisy environment. We have shown that the state-action map learned in simulation can be used in a real navigation: the vehicle exhibits a goal reaching behavior even if the current position is not clearly known. The introduction of fuzzy regions allow the selection of proper actions according to the memberships values for each region. This elementary behavior can be joined to other behaviors (e.g. obstacle avoidance, escaping from collision, and so on) in a complex navigation context. As this module depends on a vision system it has the great advantage of being reliable also for long time navigation; on the contrary, goal reaching behaviors based on information provided by odometers become unsuitable, after a while, for the cumulative errors.

Figure 7: Real paths with the fuzzy regions extension

Acknowledgements

The authors are grateful to Mr. L. Capozzo for the technical support during the experimental phase.

References

Arkin, R. 1998. *Behavior-Based Robotics.* The MIT Press, Cambridge, Massachusetts.

Asada, M.; Noda, S.; Tawaratsumida, S.; and Hosoda, K. 1996. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning* 279.

Brooks, R. 1991. Intelligence without reason. Technical report, Massachusetts Institute of Technology. A.I.Memo 1293.

Caironi, P., and Dorigo, M. 1997. Training and delayed reinforcement in q-learning agents. *International Journal of Intelligent Systems* in press.

E.Stella, and A.Distante. 1995. Self-location of a mobile robot by estimation of camera parameters. *Robotics and Autonomous System* 15.

Mahadevan, S., and Connell, J. 1992. Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence* 311–365.

Mahadevàn, S.; Theochaous, G.; and Khaleeli, N. 1998. Rapid concept learning for mobile robots. *Machine Learning.* in press.

Nayar, S., and Poggio, T. 1996. *Early Visual Learning.* Oxford University Press.

Srinivasan, M. V., and Venkatesh, S. 1997. *From living Eyes to Seeing Machines.* Oxford University Press.

Watkins, C. J., and Dayan, P. 1992. Technical note - q-learning. *Machine Learning* 8(3/4):323–339.