METHODOLOGY FOR LEARNING VISUAL REACTIVE BEHAVIOURS IN ROBOTICS THROUGH REINFORCEMENT AND IMAGE-BASED STATES

Pablo Quintía, José E. Domenech, Cristina Gamallo and Carlos V. Regueiro
Facultad de Informática, Universidad de A Coruña, 15071 A Coruña, Spain

Keywords: Reinforcement Learning, Mobile Robotics, Artificial Vision, Visual Reactive Behaviours, Motivation.

Abstract: This article describes the development of a methodology for the learning of visual and reactive behaviours using reinforcement learning. With the use of artificial vision the environment is perceived in 3D, and it is possible to avoid obstacles that are invisible to other sensors that are more common in mobile robotics. Reinforcement learning reduces the need for intervention in behaviour design, and simplifies its adjustment to the environment, the robot and the task. The designed methodology is intended to be general; thus, in order to change the desired behaviour, only the reinforcement and the filtering of the image need to be changed. For the definition of the reinforcement a laser sensor is used, and for the definition of the states a fixed 3x3 grid is used. The behaviours learned were wall following, object following, corridor following and platform following. Results are presented with a Pioneer 2 AT. A Gazebo 3D simulator was used for the Learning and testing phase, and a test of the wall following behaviour was carried out in a real environment.

1 INTRODUCTION

The design and implementation of reactive behaviours for the control of autonomous mobile robots has been shown to be one of the most efficient ways of carrying out low-level tasks. These require a very short response time and continuous interaction with the environment, which is almost totally unknown, complex and dynamic. Thus arises the challenge of specifying how each one should be implemented.

One of the most promising techniques is reinforcement learning (RL) (Millán et al., 2002; Wyatt, 1995), one of the principal advantages of which is that it minimizes interaction with the designer, since only the set of states and actions and the reinforcement has to be established. There is no need to identify all the situations in which the robot may find itself, nor the action to be implemented in each of them; only whether the result of the action is acceptable or not need be stated.

This work describes the design and implementation of four visual behaviours using a single camera, with the aim of studying the feasibility of the project and enabling a simple, economical implementation. The results obtained are generalisable, with the possible exception of the discrimination between floor and obstacles. This perception would be more robust and efficient if in-depth information were used.

We now comment on related work and go on to describe the TTD(λ) - Q-learning algorithm and its application to the four behaviours chosen. We then show the results obtained in simulation. For this phase the multirobot simulator Player/Stage/Gazebo was chosen as it is highly generalized, it supports the Pioneer 2-AT mobile robot, and simulates in 3-D (Gazebo). Lastly, we finish off with a conclusions and future work section.

2 RELATED WORK

Only a small number of studies have used vision as the principal sensorial input for visual behaviours in a mobile robot. This is probably due to the high cost associated with processing visual information (Nakamura and Asada, 1995).

In some works, reinforcement is used to learn visual behaviours that are similar to wall following (e.g.

Quintía P., E. Domenech J., Gamallo C. and V. Regueiro C. (2007).
METHODOLOGY FOR LEARNING VISUAL REACTIVE BEHAVIOURS IN ROBOTICS THROUGH REINFORCEMENT AND IMAGE-BASED STATES.
In *Proceedings of the Fourth International Conference on Informatics in Control, Automation and Robotics*, pages 149-156
DOI: 10.5220/0001628901490156

servoing and wandering), but which are simpler as there is no need to maintain a distance from the wall the robot is following. Gaskett et al. (Gaskett et al., 2000) use an improved version of Q-learning ("Advantage Learning") which handles continuous states and actions thanks to neural networks.

Another implementation of visual servoing behaviour can be found in (Martínez-Marín and Duckett, 2005). The algorithm used is another variant of Q-learning that permits real-time learning. Unlike in the present work, the state of the agent is defined by the position of the camera (inclination and angle of turn) focused on the objective, and not by the image. Thus, active vision is required, along with a perfect identification of the objective, which is not always possible. It is also difficult to use the same system to implement more than one behaviour simultaneously.

A similar, but more complete, general approach was taken in (Boada et al., 2002). This system learns basic behaviours (watching and orientation) for controlling a pan-tilt unit and the robot, and combines them to obtain complex ones (approach). Nevertheless, they need to detect and identify the objective, which is very difficult with a wall, a corridor or a platform. Once again, states were not defined directly by the image.

Ruiz et al. (Ruiz et al., 2005) have implemented two visual behaviours. Their approach is based on the detection of straight segments in the image, which to a certain degree limits its mode of use. Moreover, control is totally heuristic, with no type of learning. Our approach is more general and is not conditioned by the characteristics of the environment.

3 REINFORCEMENT LEARNING

Reinforcement learning (RL) is based on the use of a qualification (reinforcement) of the agent's actions by the environment. The reinforcement does not indicate the correct action (supervised learning), only whether it has been satisfactory or not, and is not usually immediate in time. The situations of the agent are usually codified into discrete states (s) in which various actions (a) can be implemented (may be different for each state).

A simple and intuitive definition of reinforcement has been sought, as we believe that it is one of the main advantages of this type of algorithm. Reinforcement indicates *only* those situations in which it is highly probable that the robot has ceased to implement the task correctly (i.e. the robot is excessively far from a wall), or has definitively carried it out badly (the robot collides with an element in its environment or is excessively close to one).

Learning consists of approximating a quality function $Q(s,a)$. The optimal action in each state is the one that maximizes its evaluations. The algorithm chosen for this work is *TTD(λ) - Q-learning*, due to it being based on the basic and simply *Q-Learning* algorithm but applied to several states obtaining a faster convergence time. The updating equation for the valuations is:

$$\Delta Q(s_t,a_t) = \alpha[r_t + \gamma max_a Q(s_{t+1},a) - Q(s_t,a_t)]e_t(s)\ , \quad (1)$$

for all $s_t \in S_n$ where α is the learning coefficient, γ is the reinforcement discount coefficient, S_n is the set of the n-last visited states and e_t is the eligibility trace of the state:

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & \text{if } s \neq s_t, \\ 1 & \text{if } s = s_t, \end{cases} \quad (2)$$

where λ is a decay parameter. Initial Q-values $\in [-0,9,-1,0]$.

One drawback of *Q-learning* is the need to strike a balance between exploration and exploitation. In order to do so, the *Softmax* method (Moreno et al., 2004) has been applied, where the probability of taking the action a_i in the state s at time t is:

$$Pr(s,a_i) = \frac{e^{Q_t(s,a_i)/T_s}}{\sum_{j=1}^{n} e^{Q_t(s,a_j)/T_s}}\ , \quad (3)$$

where $\{a_1,....,a_n\}$ is the set of possible actions in the state s and T_s is the temperature associated to state s.

With temperature it is possible to regulate the probability distribution between actions, and thus, the balance between exploration and exploitation. Initially we start from a high temperature (greater exploration of actions) and this is progressively reduced throughout the learning in order to principally select those actions with the best evaluation.

Temperature is modified for each state in accordance with the equation:

$$T(t) = \begin{cases} T_0\, e^{-\frac{t}{t_k} ln \frac{T_0}{k}} & \text{if } t \leq t_k, \\ k & \text{if } t > t_k, \end{cases} \quad (4)$$

where t is the number of times the current state has appeared, T_0 is the "initial" temperature, k is the minimum temperature (the state does not explore more actions) and t_k is the number of appearances that are required for the temperature to reach k.

Table 1 shows a summary of the values used for the parameters of the TTD(λ) - Q-learning algorithm.

4 BEHAVIOURS

4.1 Common Aspects

On a reactive behaviour and with RL, each state must contain all the information for selecting the next action. After several tests the camera, which uses a 80 wide-angle lens, (Fig. 1) was placed 53 cm over the robot and inclined 42 degrees towards the floor. Using these values there is no need to change the position of the camera for the different behaviours.

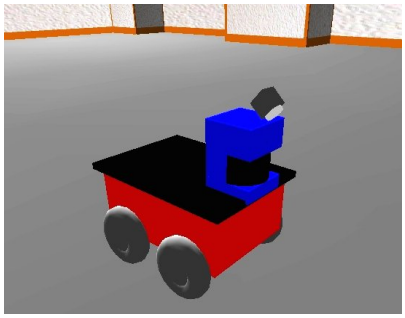


Figure 1: Position of the camera and the laser in the Gazebo simulator.

The position of the laser depends on the reinforcement wanted but, with the exception of the platform following, the laser was placed over the front of the robot, centred and parallel to the floor. For the platform following behaviour it was inclined 45 degrees towards the floor in order to detect the limits of the platform. A SICK LMS 200 laser sensor was used.

4.1.1 State Space

As was to be expected, the definition of the state space was critical in the development of this work, since it has to respond to criteria that are at times conflicting. On one hand, the space must be small, as convergence time in RL increases exponentially with the number of states. On the other hand, "perceptual aliasing" must be avoided; that is, the system should not classify two situations in which the robot must execute very different commands in the same state.

Table 1: Values used in this work for the TTD(λ) - Q-learning algorithm.

Par.	Description	Value
α	Learning coefficient	0.2
γ	Reinforcement discount coefficient	0.99
T_0	Initial temperature	0.07
k	Lower temperature limit	0.009
t_k	State exploration limit	500
λ	Decay parameter	0.9
n	Number of last visited states	3

Lastly, due to the quantity of information generated by a camera, the image needs to be processed in order to reduce the amount of pertinent information (Nehmzow, 1999). In order to resolve these problems, a simple, computationally efficient methodology has been employed, which can be run in real time on a mobile robot (Regueiro et al., 2006). This approach is easily generalisable.

Firstly, with the exception of object following, the image is processed with a Sobel edge enhancement filter (Fig. 2(b)) to highlight the pertinent information: obstacles (positive and negative) on the floor. This floor detection process is highly sensitive to changes in lighting and textures. Nevertheless, it can be improved in different ways: with stereo vision, by calibrating the camera to detect the ground plane or by applying Machine-Learning techniques for ground detection (Michels et al., 2005). For the object following behaviour the image is processed to detect a specific colour (red was chosen). Then for each pixel its red value is analysed, and if this exceeds a specified threshold it is set to white, otherwise it is set to black.

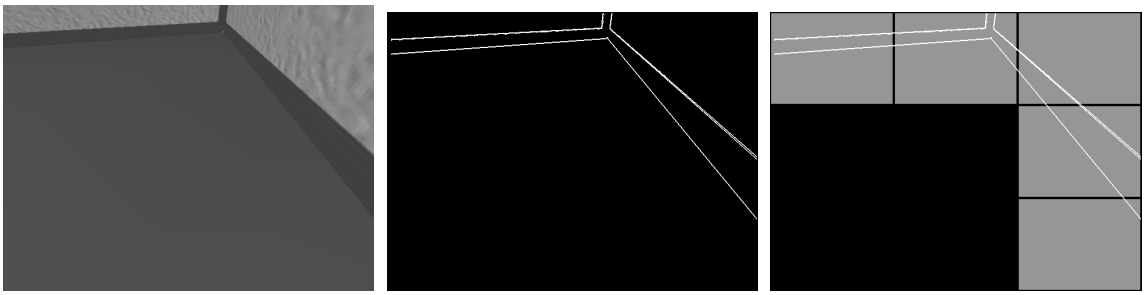
Secondly, the image is divided into a grid made up of 3 rows and 3 columns (Fig. 2(c)) for codification. A cell is considered occupied if the percentage of edge pixels reaches a given threshold. This step is essential for avoiding "perceptual aliasing". Thus defined, the state space is 2^9, and in order to reduce it, it is supposed that if a cell in one of the columns is occupied, all those cells above it are also occupied (Fig. 2(c)). Hence the number of possible states is $(3+1)^3$; i.e. 64. The state space may be further reduced, but drastic modifications would be needed in the codification, which would be difficult to justify.

4.1.2 Action Space

The action space was chosen in order to allow a flexible behaviour, and at the same time avoid a series of actions that would leave the robot immobile and without negative reinforcement. One constraint of Q-learning is that actions must be discrete. Hence, the action space chosen is as in Table 2

4.2 Wall Following Behaviour

It has been shown that wall following behaviour is one of the most useful when robots need to move reactively and safely through their environment (Regueiro et al., 2002). One of its advantages is that it only uses local information, and it makes use of the topological structure of the environment.



(a) (b) (c)

Figure 2: Determination of state with an image: a) original image; b) edge pixels (Sobel filter); and c) final codified state (showing the codification grid and the free and occupied cells).

4.2.1 Definition of Reinforcement

The defined reward is always negative (-1), spurious in time, and has two different components:

1. Close reinforcement: if any of the laser beams detects an obstacle at 25 cm or less.
2. Far reinforcement: if the right-hand beam detects a distance greater of 1 m.

The reinforcement has been defined in two ways:

1. Instant reinforcement: the reinforcement is applied when it finds a non desired situation. Used with the close reinforcement.
2. Delayed reinforcement: the reinforcement is applied when a undesired situation persists over a period of time. Used with the "far reinforcement".

With delayed reinforcement the need to always have a wall on the right side is relaxed and thus a more robust behaviour is possible.

Table 2: Action space chosen.

Act. n.	Linear speed m/s	Angular speed rad/s
0	0.4	0.2
1	0.4	0
2	0.4	-0.2
3	0.2	0.8
4	0.2	0.6
5	0.2	0.4
6	0.2	0.2
7	0.2	0
8	0.2	-0.2
9	0.2	-0.4
10	0.2	-0.6
11	0.2	-0.8
12	0.05	0.8
13	0.05	-0.8

4.2.2 Results

The environment used is shown in Fig. 4(a).

Wall following behaviour belongs to the class of continuous tasks which persist over time. This means that they are not divided naturally into episodes, as would be desirable for application of a reinforcement algorithm. In order to avoid this drawback a reinforcement is generated after each serious system error (collision or excessive distancing) and the robot executes a recovery function which returns the robot to a safe state.

Fig. 3 shows the reinforcement received during the learning phase in TTD(λ). Each point represents the reinforcement accumulated in the previous 400 learning cycles. Thus, the first point indicates that the reinforcement received since the onset of the experiment up until 400 cycles has been -12, and so forth. The learned Q-values are stored for their subsequent testing.

As can be seen in the diagrams, the agent learns the task in 2,000 learning cycles with TTD(λ) as opposed to 80,000 cycles obtained with Q-Learning algorithm and 8x4 grid (Regueiro et al., 2006).

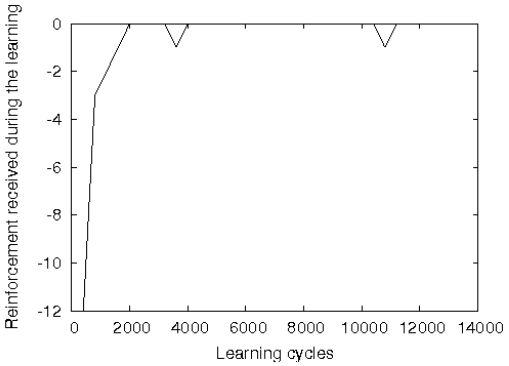
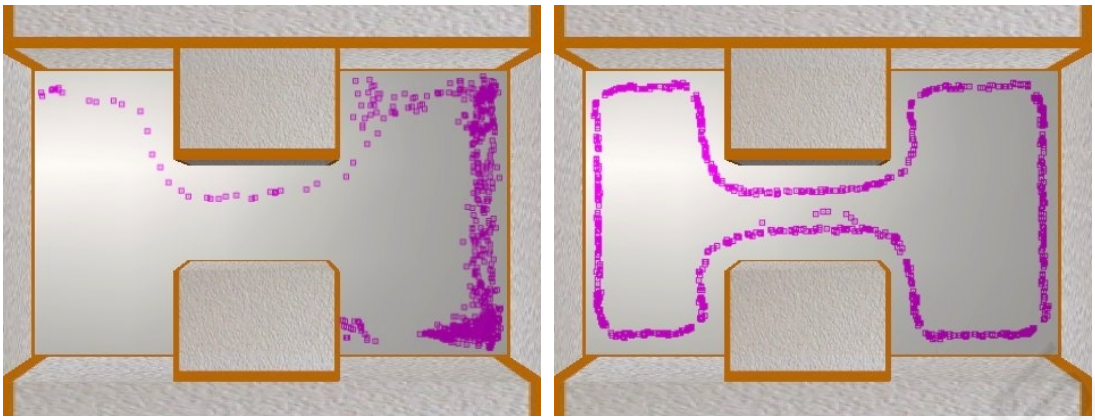


Figure 3: Reinforcement accumulated during the learning of wall following behaviour.



(a) (b)

Figure 4: Wall following behaviour: a) Trajectories obtained during initial stages; and b) Trajectories obtained during final stages.

4.3 Corridor Following Behaviour

The corridor following behaviour is one of the simplest behaviours to learn; nevertheless, when working with RL it has problems with decision selection at crossroads. It simply makes no decision, as could be expected from a reactive behaviour, thus the robot cannot be ordered to go straight on or to turn at a junction. Several positive reinforcements were used to make the robot go straight on, but without success.

4.3.1 Definition of Reinforcement

Two classes of reinforcements were defined:

1. Critical reinforcement: if any of the laser beams detects an obstacle at 25 cm or less the reward is -1.
2. Speed reinforcement: used to favour the use of straight and fast actions or/and penalise slow and turning actions with values in $[-0.4, 0.3]$.

There were no significant variations in the final behaviours with the use of speed reinforcement. We have to mention that even when defining an erroneous reinforcement the robot learned the correct behaviour.

4.3.2 Results

The environment used is shown in Fig. 6(a).

As has previously been mentioned, the final behaviours show a tendency to turn to right or left. The example in Fig. 6(b) shows a tendency to turn left; if not to straight on and then turn right. That is the reason for the difference between turns to the left, where the robot turns smoothly and passes close to the left wall, and the right turns, where the robot tries to go straight until the wall is directly in front of it.

Fig. 5 shows the reinforcement received during the learning phase in TTD(λ). As can be seen in the diagrams, the agent learns the task in 800 learning cycles.

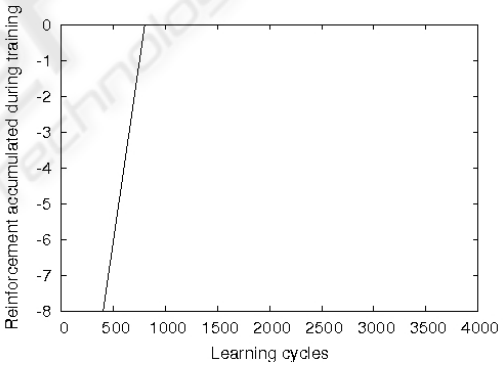


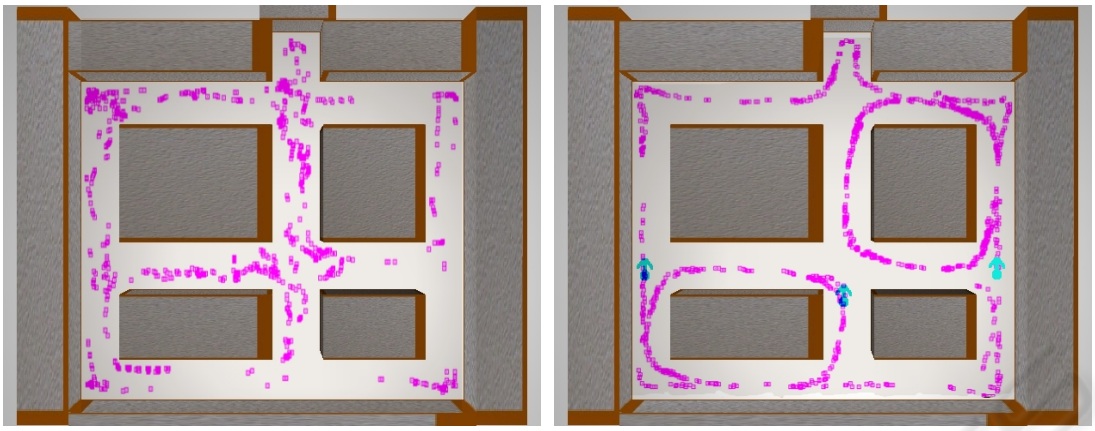
Figure 5: Reinforcement accumulated during the learning of corridor following.

4.4 Platform Following Behaviour

The platform following behaviour is very similar to the corridor following behaviour, but the laser sensor points 45 degrees towards the ground, as it has to detect uneven features in the floor. This behaviour also has the same problem at crossroads.

4.4.1 Definition of Reinforcement

A simple calibration of the laser is needed at the beginning of the execution. The laser measures the distance of the front beam (number 90) and this distance is used to calculate the expected distance for the rest of the beams (values between 50 and 130). If a laser



(a) (b)

Figure 6: Corridor following behaviour: a) Trajectories obtained during initial stages; and b) Trajectories obtained during final stages.

beam value is greater than the expected distance plus 15 cm, then the reward is -1.

4.4.2 Results

The environment used is shown in Fig. 8(a).

Fig. 7 shows the reinforcement received during the learning phase in TTD(λ). As can be seen in the diagrams, the agent learns the task in 2,000 learning cycles.

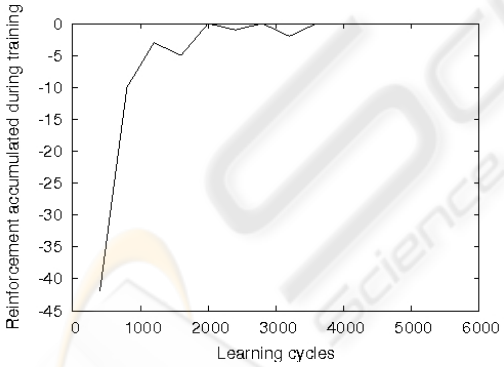


Figure 7: Reinforcement accumulated during the learning of platform following behaviour.

4.5 Object Following Behaviour

The main change for this behaviour was the filtering of the image. Here, instead of using a Sobel filter the image is preprocessed with a simple colour filter. Red was chosen and a red box is attached to the prey, another Pioneer 2-AT executing a follow line behaviour. The prey has a follow line behaviour, the path of the pray is shown in Fig. 10(a).

4.5.1 Definition of Reinforcement

The defined reward is negative (-1.0) when any of the following conditions is accomplished:

1. Close reinforcement: if any of the laser beams detects an obstacle at 30 cm or less.
2. Image reinforcement: if the prey is not viewed by the camera.

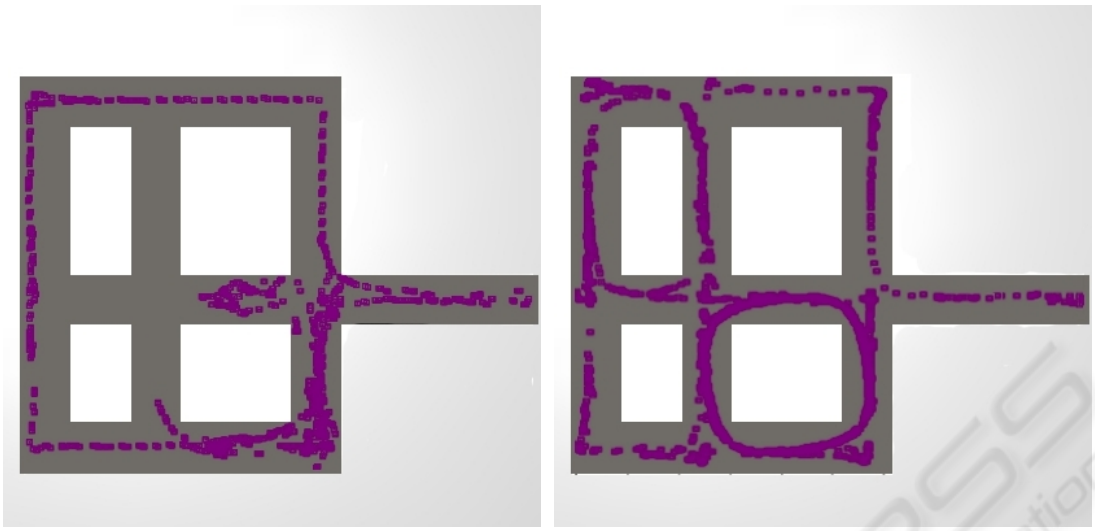
4.5.2 Results

Fig. 9 shows the reinforcement received during the learning phase in TTD(λ). As can be seen in the diagrams, the agent learns the task in 10,500 learning cycles. The learning is slower for this behaviour due to the fact that the path of the prey is very long, an entire lap taking approximately 7 minutes, so a number of difficult situations appeared very sporadically.

As can be seen in Fig. 10(b) the path of the final behaviour is smoother than the path of the prey. The average distance between the robot and a prey moving at constant speed of 0.2 m/s is 0.9 m. We have tested several speeds for the prey: if the speed is slower than 0.35 m/s the robot can follow it (it should be remembered that the maximum speed of the robot is 0.4 m/s, see table 2).

4.6 Test in a Real Robot

Finally we have tested the behaviour with minor changes on a real environment: a long corridor, narrower than learning environment, with many new features, such as trash baskets, radiators, plants, glass walls, open and closed doors. This real environment



(a) (b)

Figure 8: Platform following behaviour: a) Trajectories obtained during initial stages; and b) Trajectories obtained during final stages.

(a) (b)

Figure 10: Object following behaviour: a) Trajectories obtained during initial stages; and b) Trajectories obtained during final stages.

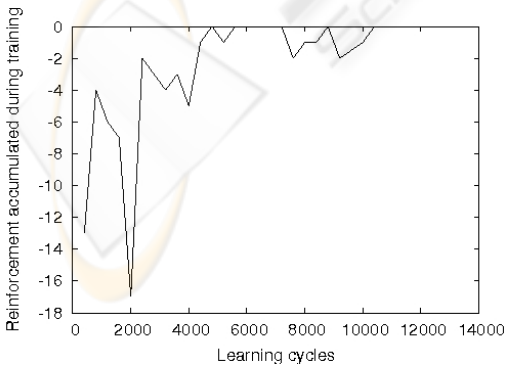


Figure 9: Reinforcement accumulated during the learning of object following behaviour.

also has both natural and artificial lighting. As can be seen by examining the robot's path on Fig. 11, our behaviour works perfectly.

5 CONCLUSIONS AND FUTURE WORK

In this work four visual reactive behaviours for the Pioneer 2 AT robot have been implemented with RL algorithms. All behaviours use the same state and action spaces, only reinforcement changes. A rangefinder laser is used to define the reinforcement of each behaviour. Learning and testing were carried

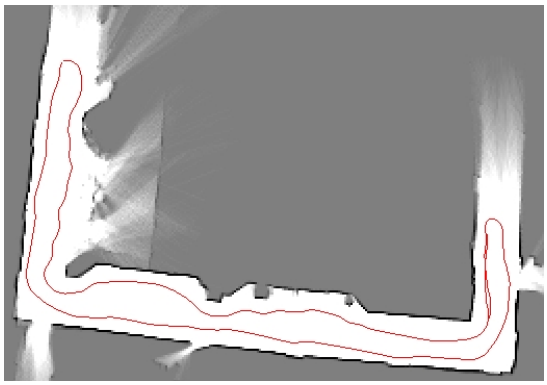


Figure 11: Robot's path carrying out the visual wall following behaviour on a real environment (right wall). The map is a certainty grid created with a LMS200 laser (white pixels are free space, black obstacles and grey uncertainty).

out on the 3-D Gazebo simulator. A wall following test in a real environment is also shown.

Using a fisheye camera the environment is perceived in 3D, and it is possible to avoid obstacles that are invisible to other sensors which are more common in mobile robotics (laser or ultrasounds).

In the proposed methodology the states are defined directly from the image after a simple preprocessing (Sobel or colour filtering) with no calibration process. With a 3x3 grid, we can define a state space of only 64 states. It has a certain degree of "perceptual aliasing", but RL algorithm converges. We have also tested grids of different dimensions with similar results but greater convergence time. A delicate balance need be struck between reducing the number of states and avoiding "perceptual aliasing".

The proposed codification and methodology is general, not specific for the task, and has proved to be efficient and valid, and easy to adapt to distint behaviours. The system works with different types of reinforcement and filtering.

Various tests were carried out to verify the robustness of the learned behaviours. We used obstacles that were not detected by the laser device, and walls with gaps. In both cases the system generalized perfectly and the results were optimal. If the gaps in the walls were large (over 40 cm) a large number of new states appeared with respect to the training process, and the final result deteriorated.

Future lines of work include on-line real robot learning, the integration of several behaviours (e.g. follow objects and avoid obstacles) and establishing a mechanism for automatically defining the states of RL (e.g. neural networks).

ACKNOWLEDGEMENTS

This paper was supported in part by the Xunta de Galicia and Spanish Government under Grants PGIDIT04-TIC206011PR, TIN2005-03844, and department colaboration grant (B.O.E. 16-06-2006).

REFERENCES

Boada, M., Barber, R., and Salichs, M. (2002). Visual approach skill for a mobile robot using learning and fusion of simple skills. *Robotics and Autonomous Systems*, 38:157–170.

Gaskett, C., Fletcher, L., and Zelinsky, A. (2000). Reinforcement learning for a vision based mobile robot. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, volume 1, pages 403–409.

Martínez-Marín, T. and Duckett, T. (2005). Fast reinforcement learning for vision-guided mobile robots. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 4170–4175.

Michels, J., Saxena, A., and Ng, A. (2005). High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proc. Int. Conf. on Machine Learning*.

Millán, J. R., Posenato, D., and Dedieu, E. (2002). Continuous-action Q-Learning. *Machine Learning*, 49:247, 265.

Moreno, D., Regueiro, C., Iglesias, R., and Barro, S. (2004). Using prior knowledge to improve reinforcement learning in mobile robotics. In *Towards Autonomous Robotic Systems (TAROS)*.

Nakamura, T. and Asada, M. (1995). Motion sketch: Acquisition of visual motion guided behaviors. In *IJCAI*, pages 126–132.

Nehmzow, U. (1999). Vision processing for robot learning. *Industrial Robot*, 26(2):121–130.

Regueiro, C., Domenech, J., Iglesias, R., and Correa, J. (2006). Acquiring contour following behaviour in robotics through q-learning and image-based states. In *Proc. XV Int. Conf. on Computer and Information Sciences Engineering (CISE)*, pages 403–409.

Regueiro, C., Rodríguez, M., Correa, J., Moreno, D., Iglesias, R., and Barro, S. (2002). A control architecture for mobile robotics based on specialists. In Leondes, C., editor, *Intelligent Systems: Technology and Applications*, volume 6, pages 337–360. CRC Press.

Ruiz, J., Montero, P., Martín, F., and Matellán, V. (2005). Vision based behaviors for a legged robot. In *Proc. Workshop en Agentes Físicos (WAF)*, pages 59–66.

Wyatt, J. (1995). Issues in putting reinforcement learning onto robots. In *10th Biennal Conference of the AISB*, Sheffield, UK.