Gradient Based Reinforcement Learning for Autonomous Underwater Cable Tracking

Andres El-Fakdi and **Marc Carreras** and **Emili Hernandez**[1]

Abstract. This paper proposes a field application of a high-level Reinforcement Learning (RL) control system for solving the action selection problem of an autonomous robot in a cable tracking task. The learning system is characterized by using a policy gradient based search method for learning the internal state/action mapping. The function approximator used to represent the policy is a barycentric interpolator. Policy only algorithms may suffer from long convergence times when dealing with real robotics. In order to speed up the process, the learning phase has been carried out in a simulated environment using the hydrodynamic model of the vehicle and, in a second step, the policy has been transferred and tested successfully on a real robot. Future steps plan to continue the learning process on-line while on the real robot while performing the mentioned task. We demonstrate its feasibility with real experiments on the underwater robot ICTINEU Autonomous Underwater Vehicle (AUV).

1 INTRODUCTION

Reinforcement Learning (RL) is a widely used methodology in robot learning, see [23]. In RL, an agent tries to maximize a scalar evaluation obtained as a result of its interaction with the environment. The goal of a RL system is to find an optimal policy to map the state of the environment to an action which in turn will maximize the accumulated future rewards. The agent interacts with a new, undiscovered environment selecting actions for each state, receiving a numerical reward for every decision. Obtained rewards are used to teach the agent so the robot learns which action to take at each state, achieving an optimal or sub-optimal policy (state-action mapping).

The dominant approach over the last decade has been to apply reinforcement learning using the value function approach. Although value function methodologies have worked well in many applications, they have several limitations. The considerable amount of computational requirements that increase time consumption and the lack of generalization among continuous variables represent the two main disadvantages of "value" RL algorithms. Over the past few years, studies have shown that approximating a policy can be easier than working with value functions, and better results can be obtained ([24] [2]). As presented in [1], it is intuitively simpler to determine *how to act* instead of *value of acting*. So, rather than approximating a value function, new methodologies approximate a policy using an independent function approximator with its own parameters, trying to maximize the future expected reward. Only a few but promising practical applications of policy gradient algorithms have appeared, this paper emphasizes the work presented in [5], where an autonomous helicopter learns to fly using an off-line model-based policy search method. Also important is the work presented in [21] where a simple "biologically motivated" policy gradient method is used to teach a robot in a weightlifting task. More recent is the work done in [10] where a simplified policy gradient algorithm is implemented to optimize the gait of Sony's AIBO quadrupedal robot. More recently, the work presented in [18] gives an overview on learning with policy gradient methods for robotics while presenting the results obtained in the application of hitting a baseball with an anthropomorphic arm.

All these recent applications share a common drawback, gradient estimators used in these algorithms may have a large variance (see [13] and [11]) what means that policy gradient methods learn much more slower than RL algorithms using a value function (see [24]) and they can converge to local optima of the expected reward (see [15]), making them less suitable for on-line learning in real applications. In order to decrease convergence times and avoid local optima, newest applications combine policy gradient algorithms with other methodologies, it is worth to mention the work done in [25] and [14], where a biped robot is trained to walk by means of a "hybrid" RL algorithm that combines policy search with value function methods.

A good proposal for speeding up gradient methods may be offering the agent an initial policy. Example policies can direct the learner to explore the promising part of search space which contains the goal states, specially important when dealing with large state-spaces whose exploration may be infeasible. Also, local maxima dead ends can be avoided with example techniques [12]. The idea of providing high-level information and then use machine learning to improve the policy has been successfully used in [22] where a mobile robot learns to perform a corridor following task with the supply of example trajectories. In [4] the agent learns a reward function from demonstration and a task model by attempting to perform the task. Finally, the work done in [9] concerning an outdoor mobile robot that learns to avoid collisions by observing a human driver operating the vehicle.

This paper proposes a reinforcement learning application where the underwater vehicle $ICTINEU^{AUV}$ carries out a visual based cable tracking task using a direct gradient algorithm to represent the policy. The function approximator used to represent the policy a barycentric interpolator function. An initial example policy is first computed by means of computer simulation where a hydrodynamic model of the vehicle simulates the cable following task. Once the simulated results are accurate enough, in a second phase, the policy is transferred to the vehicle and executed in a real test. A third step will be mentioned as a future work, where the learning procedure continues on-line while the robot performs the task, with the objective of improving the initial example policy as a result of the interaction with the real environment. This paper is structured as follows. In Section 2 the learning procedure and the policy gradient algorithm are detailed. Section 3 describes all the elements that affect our problem: the underwater robot $ICTINEU^{AUV}$, the mathematical model of

[1] University of Girona, Spain, email: aelfakdi@eia.udg.edu

the vehicle used in the simulation, the vision system and the controller. Details and results of the simulation process and the real test are given in Section 4 and finally, conclusions and the future work to be done are included in Section 5.

2 LEARNING PROCEDURES

The introduction of prior knowledge in a gradient descent methodology can dramatically decrease the convergence time of the algorithm. This advantage is even more important when dealing with real systems, where timing is a key factor. Such learning systems can divide its procedure into two phases or steps as shown in Fig. 1. In the first phase of learning (see Fig. 1(a)) the learner interacts with a simulated environment; during this phase, the agent extracts all useful information from simulation. In a second step, once it is considered that the agent has enough knowledge to build a "secure" policy, it takes control of the real robot and the learning process continues in the real world, see Fig. 1(b).

The proposal presented here takes advantage of learning by simulation as an initial startup for the learner. The objective is to transfer an initial policy, learned in a simulated environment, to a real robot and test the behavior of the learned policy in real conditions. First, the learning task will be performed in simulation with the aid of the hydrodynamic model of the robot. Once the learning process is considered to be finished, the policy will be transferred to $ICTINEU^{AUV}$ in order to test it in the real world. In a future task, the learning procedure will switch to a third phase, continuing to improve the policy while in real conditions. The Baxter and Bartlett approach [6] is the gradient descent method selected to carry out the simulated learning corresponding to phase one. Next subsection gives details about the algorithm.

2.1 The gradient descent algorithm

The Baxter and Bartlett's algorithm is a policy search methodology

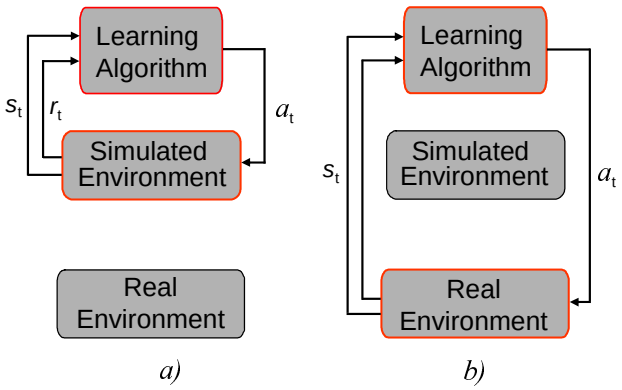


Figure 1. Learning phases.

The algorithm works as follows: having initialized the parameters vector θ_0, the initial state i_0 and the eligibility trace $z_0 = 0$, the learning procedure will be iterated T times. At every iteration, the parameters' eligibility z_t will be updated according to the policy gradient approximation. The discount factor $\beta \in [0, 1)$ increases or decreases the agent's memory of past actions. The immediate reward received $r(i_{t+1})$, and the learning rate α allows us to finally compute the new vector of parameters θ_{t+1}. The current policy is directly modified by the new parameters becoming a new policy to be followed by the next iteration, getting closer to a final policy that represents a correct solution of the problem.

Algorithm 1: Baxter and Bartlett's OLPOMDP algorithm

1. Initialize:
   $T > 0$
   Initial parameter values $\theta_0 \in R^K$
   Initial state i_0
2. Set $z_0 = 0$ ($z_0 \in R^K$)
3. $for\ t = 0$ to T do:
   (a) Observe state x_t
   (b) Generate control action u_t according to current policy $\mu(\theta, x_t)$
   (c) Observe the reward obtained $r(i_{t+1})$
   (d) Set $z_{t+1} = \beta z_t + \frac{\nabla \mu_{u_t}(\theta, x_t)}{\mu_{u_t}(\theta, x_t)}$
   (e) Set $\theta_{t+1} = \theta_t + \alpha_t r(i_{t+1}) z_{t+1}$
4. end

The algorithm is designed to work on-line. Our policy will be approximated with a particular class of functions called the *barycentric interpolators* (see [16]), which use an interpolation process based on a finite set of discrete points conforming a mesh. This mesh does not need to be regular, but the method outlined here assumes that the state space is divided into a set of rectangular boxes as shown in Fig. 2. The key here is that any particular point is enclosed in a rectangular box that can be defined by 2^N nodes in our N-dimensional state space.

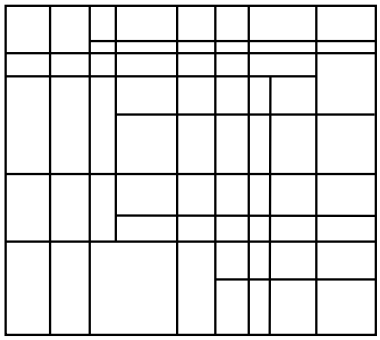


Figure 2. Example of a 2-dimension rectangular mesh.

Let $\Sigma^\delta = \{\xi_i\}_i$ be a set of points distributed in the mesh at some resolution δ on the state of dimension d. For any state x inside some rectangular box $(\xi_i, ..., \xi_n)$, x is the *barycenter* of the $\{\xi_i\}_{i=1..n}$ inside this box with positive coefficients $p(x|\xi_i)$ of sum 1 called the *barycentric coordinates* (see Fig. 3) where:

$$x = \sum_{i=1..n} p(x|\xi_i)\xi_i \qquad (1)$$

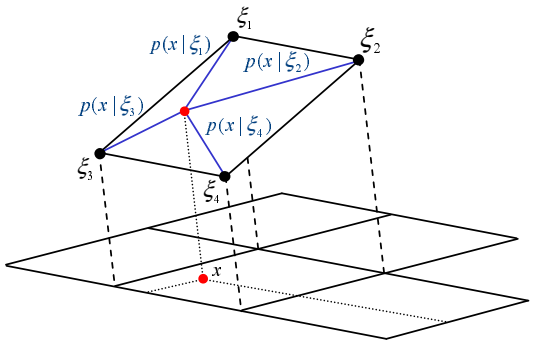


Figure 3. Graphic representation of the *barycentric coordinates* given a state x in a 2 dimensional mesh case.

We can set $V^{\delta}(\xi_i)$ as the value of the function at the points previously described ξ_i. As seen in Eq. 2, $V^{\delta}(x)$ is the **barycentric interpolator** of state x which is the barycenter of the points $\{\xi_i\}_{i=1..n}$ for some box $(\xi_i, ..., \xi_n)$ with barycentric coordinates $p(x|\xi_i)$, see Fig. 4.

$$V^{\delta}(x) = \sum_{i=1..n} p(x|\xi_i) V^{\delta}(\xi_i) \tag{2}$$

As stated before, our policy will be directly approximated using a barycentric interpolator function whose values $V^{\delta}(\xi_i)$ represent the policy parameters. Therefore, given and input state x_t the policy will compute a continuous control action $V^{\delta}(x)_t = u_t$ driving the learner to a new state with its associated reward. Once the action has been selected, the parameter update process starts. The barycentric interpolator parameters are updated following expres
Algorithm 1:

$$z(\xi_i)_{t+1} = \beta z(\xi_i)_t + \frac{\nabla \mu_{u_t}(V^{\delta}(\xi_i), x_t)}{\mu_{u_t}(V^{\delta}(\xi_i), x_t)}$$

$$z(\xi_i)_{t+1} = \beta z(\xi_i)_t + p(x|\xi_i)e$$

the error at the output is given by:

$$e = V^{\delta}(x)_{desired} - V^{\delta}(x)$$

$$z(\xi_i)_{t+1} = \beta z(\xi_i)_t + p(x|\xi_i)(V^{\delta}(x)_{desired} - V$$

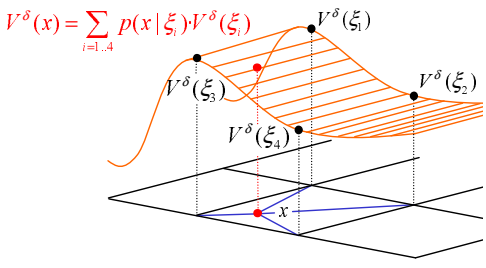


Figure 4. Calculation of the function approximator outpu
particular state x.

Finally, the old parameters are updated following expression 3.(e) of Algorithm 1:

$$V^{\delta}(\xi_i)_{t+1} = V^{\delta}(\xi_i)_t + \alpha r(i_{t+1}) z_{t+1} \tag{7}$$

The vector $V^{\delta}(\xi_i)$ represents the policy parameters to be updated, $r(i_{t+1})$ is the reward given to the learner at every time step, z_{t+1}describes the estimated gradients mentioned before and, at last, we have α as the learning rate of the algorithm.

3 CASE TO STUDY: CABLE TRACKING

This section is going to describe the different elements that take place into our problem: first, a brief description of the underwater robot $ICTINEU^{AUV}$ and its model used in simulation is given. The section will also present the problem of underwater cable tracking and, finally, a description of the barycentric interpolator function designed for both, the simulation and the real phases is detailed.

3.1 $ICTINEU^{AUV}$

The underwater vehicle $ICTINEU^{AUV}$ was originally designed to compete in the SAUC-E competition that took place in London during the summer of 2006 [19]. Since then, the robot has been used as a research platform for different underwater inspection projects which include dams, harbors, shallow waters and cable/pipeline inspection.

The main design principle of $ICTINEU^{AUV}$ was to adopt a cheap structure simple to maintain and upgrade. For these reasons, the robot has been designed as an open frame vehicle. With a weight

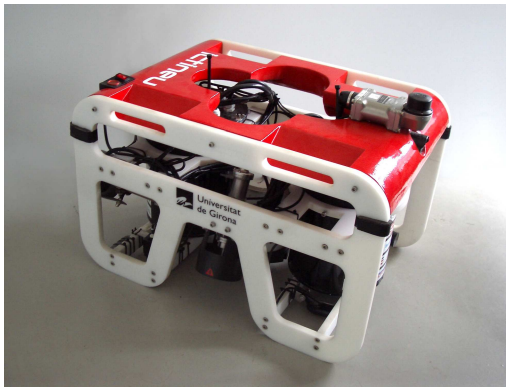


Figure 5. The autonomous underwater vehicle $ICTINEU^{AUV}$.

3.2 AUV mathematical model

As described in the literature [8], the non-linear hydrodynamic equation of motion of an underwater vehicle with 6 *degrees of freedom* (DOF), in the body fixed frame, can be conveniently expressed as:

$$\tau^B + G(\eta) - D(\upsilon^B)\upsilon^B + \tau_p = (M^B_{RB} + M_A)\dot{\upsilon}^B + ... \\ ... + (C^B_{RB}(\upsilon^B) + C_A(\upsilon^B))\upsilon^B \quad (8)$$

Where:

- υ^B is the velocity vector.
- $\dot{\upsilon}^B$ is the acceleration vector.
- $\eta = (\phi\theta\psi)^T$ are the Roll, Pith and Yaw angles.
- τ^B are the forces and moments exerted by thrusters.
- $G(\eta)$ are the gravity and buoyancy forces.
- $D(\upsilon^B)$ are the linear and quadratic damping matrixes.
- $D(\tau_P$ are the not modeled perturbations.
- M^B_{RB} is the inertia matrix.
- M^B_A is the added mass matrix.
- C^B_{RB} is the rigid body Coriolis and centripetal matrix.
- C^B_A is the hydrodynamic Coriolis and centripetal matrix.

Identification of the complete set of coefficients and hydrodynamic derivatives which appear in Eq. 8 is a rather complex task. The identification problem can be much more easily approached if the following simplifications are applied:

- $D(\upsilon^B)$ consists of the lineal and quadratic damping forces and can be assumed diagonal.
- M^B_{RB} and M^B_A can be assumed diagonal (this is true for $ICTINEU^{AUV}$ due to its squared shape, see Subsection 3.1).
- The body frame is located at the gravity center.

Moreover, if the robot is actuated in a single DOF during the identification experiments, further simplifications can be carried out. For instance, let's consider the dynamic equation for the surge (movement along X axis) DOF:

$$\tau_u + (\sin(\theta)B - \sin(\theta)W) - (X_u + X_{u|u|}|u|)u + ... \\ ... + \tau_P = (m - X_{\dot{u}})\dot{u} + [(m - Z_{\dot{w}})wq] - [(m - Y_{\dot{v}})vr] \quad (9)$$

which follows the standard notation proposed in [8]. If we excite the robot in a single DOF, surge in this case, in such a way that:

- $u \neq 0$ and $v = w = p = q = r = 0$
- $\theta = \phi = 0$

uncoupled experiment can be achieved, Eq. 9 can be rewritten as:

$$\dot{u} = -\frac{X_u}{(m - X_{\dot{u}})}u - \frac{X_{u|u|}|u|}{(m - X_{\dot{u}})}u + \frac{\tau_u}{(m - X_{\dot{u}})} + \frac{\tau_P}{(m - X_{\dot{u}})} \quad (10)$$

The same procedure can be applied to each degree of freedom so we can consider a generic uncoupled equation of motion for the i-degree of freedom as:

$$\dot{x}_i = \alpha_i x_i + \beta_i x_i|x_i| + \gamma_i\tau_i + \delta_i \quad (11)$$

where the state variable x represents speed. Hence, things become easier if we use Eq. 11 for the identification. Once the whole model has been uncoupled, its parameters have been identified by means of parameter identification methods [20]. The resultant model has been reduced to emulate a robot with only three degrees of freedom (DOF), X movement, Y movement and rotation respect Z axis. The identified parameters can be see in Table 1. Note that related to the squared shape of the vehicle, the robot behavior has been considered equal in X and Y; also, the β term corresponding to the quadratic damping has been neglected due to the low speeds achieved by $ICTINEU^{AUV}$ during the tests.

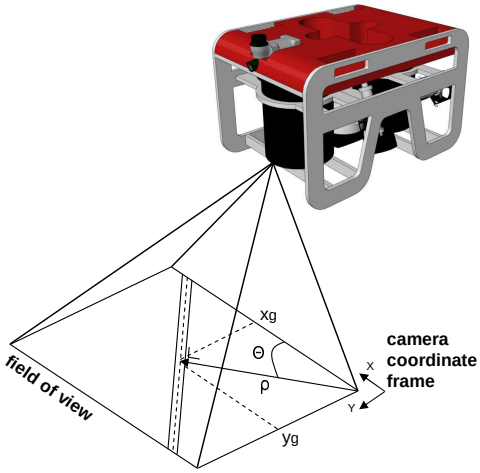


Figure 6. Coordinates of the target cable with respect $ICTINEU^{AUV}$.

Table 1. Parameter identification results.

	Parameters			
DOF	α	β	γ	δ
Surge (X movement)	0.3222	0	0.0184	0.0012
Sway (Y movement)	0.3222	0	0.0184	0.0012
Yaw (Z rotation)	1.2426	0	0.5173	-0.050

3.3 The Cable Tracking Vision System

The downward-looking B/W camera installed on $ICTINEU^{AUV}$ will be used for the vision algorithm to track the cable. It provides a large underwater field of view (about 57° in width by 43° in height). This kind of sensor will not provide us with absolute localization information but will give us relative data about position and orientation of the cable respect to our vehicle: if we are too close/far or if we should move to the left/right in order to center the object in our image. The vision-based algorithm used to locate the cable was first proposed in [17] and later improved in [3]. It exploits the fact that artificial objects present in natural environments usually have distinguishing features; in the case of the cable, given its rigidity and shape, strong alignments can be expected near its sides. The algorithm will evaluate the polar coordinates ρ (orthogonal distance from the origin of the camera coordinate frame) and Θ (angle between ρ and X axis of the camera coordinate frame) of the straight line corresponding to the detected cable in the image plane (see Fig. 6).

Once the cable has been located and the polar coordinates of th corresponding line obtained, as the cable is not a thin line but a larg rectangle, we will also compute the cartesian coordinates (x_g,y_g (see Fig. 6) of the cable's centroid with respect to the image plan by means of (12).

$$\rho = xcos(\Theta) + ysin(\Theta) \quad (12)$$

where x and y correspond to the position of any point of the lin in the image plane. The computed parameters x_g and Θ togeth with its derivatives $\frac{\delta x_g}{\delta t}$ and $\frac{\delta \Theta}{\delta t}$ will conform the input state to ou policy function represented by the barycentric interpolator. For th simulated phase, a downward-looking camera model has been use to emulate the vision system of the vehicle.

3.4 Design of the barycentric interpolator to represent the policy

As stated in previous section, the observed state is a 4 dimension vec tor $x = (x_g, \Theta, \frac{\delta x_g}{\delta t}, \frac{\delta \Theta}{\delta t})$. The x component of the cable centroid i the image plane x_g is ranged from $[0, 1.078]$ meters, $\Theta = [\frac{-\pi}{2}, \frac{\pi}{2}$ radians, the derivative of $x_g, \frac{\delta x_g}{\delta t} = [-0.5, 0.5] m/s$ and finall $\frac{\delta \Theta}{\delta t} = [-1, 1] rad/s$. From these observations, the robot will tak decisions concerning two degrees of freedom, Y movement (Sway and Z axis rotation (Yaw), therefore the continuous action vector defined as $u = (u_{sway}, u_{yaw})$ where $u_{sway} = [-1, 1] m/s$ and $u_{yaw} = [-1, 1] rad/s$. The X movement of the vehicle (surge) will not be learned. A simple controller has been implemented to control the X DOF; only if the cable is centered in the image plane the robot will move forward ($u_{surge} = 0.3 m/s$), otherwise $u_{surge} = 0$.

In order to decrease the function approximator complexity reducing the number of grid points, the main policy has been split into two subpolicies, each one represented by a 2-dimensional barycentric interpolator. It can be easily noticed that u_{sway} actions will be mainly affected by the position and the velocity of x_g along the X axis of the image plane. In the same way, u_{yaw} actions will strongly depend on the angle Θ of the cable in the image plane. Although learning uncoupled policies will probably reduce the final performance of the learner, it has been a good initial startup to focus the problem. In Fig. 7, the observed substate (x_g, $\frac{\delta x_g}{\delta t}$) is the input of subpolicy a), being the output u_{sway}. Subpolicy b) has (Θ, $\frac{\delta \Theta}{\delta t}$) as input variables and u_{yaw} as output. The density factor δ of the barycentric mesh for both grids has been experimentally set to 10 equal divisions for each axis, therefore the mesh has 100 cells.

4 RESULTS

4.1 First phase: Simulated Learning

The model of the underwater robot $ICTINEU^{AUV}$ navigates a two dimensional world at 1 meter height above the seafloor. The simulated cable is placed at the bottom in a fixed circular position. The learner has been trained in an episodic task. An episode ends either every 15 seconds (150 iterations) or when the robot misses the cable in the image plane, whatever comes first. When the episode ends, the robot position is reset to a random position and orientation around the cable's location, assuring any location of the cable within the image plane at the beginning of each episode. According to the values of the state parameters θ and x_g, a scalar immediate reward is given each iteration step. Three values were used: -10, -1 and 0. In order to maintain the cable centered in the image plane, the non negative reward $r = 0$ is given when the position along the X axis of the

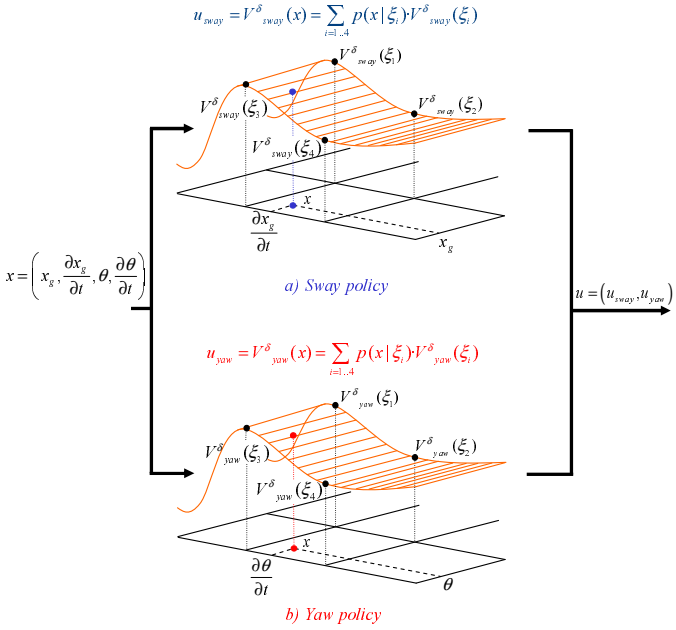


Figure 7. Representation of both subpolicies. a) represents the sway policy. b) represents the yaw policy.

centroid (x_g) is around the center of the image ($x_g \pm 0.15$) and the angle θ is close to 90° ($90^\circ \pm 15^\circ$). A $r = -1$ is given in any other location within the image plane. The reward value of -10 is given when the vehicles misses the target and the episode ends.

The number of episodes to be done has been set to 2000. For every episode, the total amount of reward perceived is calculated. Fig. 8 represents the performance of the computed policy as a function of the number of episodes when trained using Baxter and Bartlett's algorithm. The experiment has been repeated in 100 independent runs. The results here presented are the mean over these runs. The learning rate was set to $\alpha = 0.001$ and the discount factor $\beta = 0.98$. In Fig. 9 and Fig. 10 we can observe a state/action mapping of a trained agent in both, yaw and sway degrees of freedom. Figure 11 represents the trajectory once the training period finishes.

4.2 Second phase: Learned policy transfer. Real test

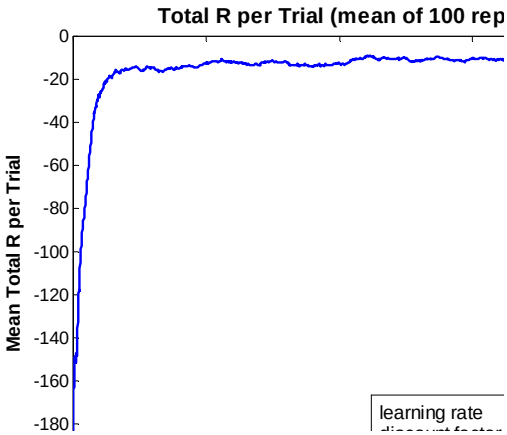
Once the learning process is considered to be finished, the resultant policy is transferred to $ICTINEU^{AUV}$ and its performance tested in a real environment. The experimental setup can be seen in Fig. 12 where the detected cable is shown while the vehicle performs a test inside the pool. Fig. 13 represents real measured trajectories of the θ angle while the vehicle performs different attempts to center the cable in the image.

5 CONCLUSIONS AND FUTURE WORKS

This paper proposes a field application of a high-level Reinforcement Learning (RL) control system for solving the action selection problem of an autonomous robot in cable tracking task. The learning system is characterized by using a direct policy search algorithm for robot control based on Baxter and Bartlett's direct-gradient algorithm. The policy is represented by 2 barycentric interpolators with

2 input state variables, each one controlling one degr
In order to speed up the process, the learning phase
ried out in a simulated environment and then transfe
successfully on the real robot $ICTINEU^{AUV}$.

Results of this work show a good performance of t
icy. Although it is not a hard task to learn in simulati
the learning autonomously in a real situation represe



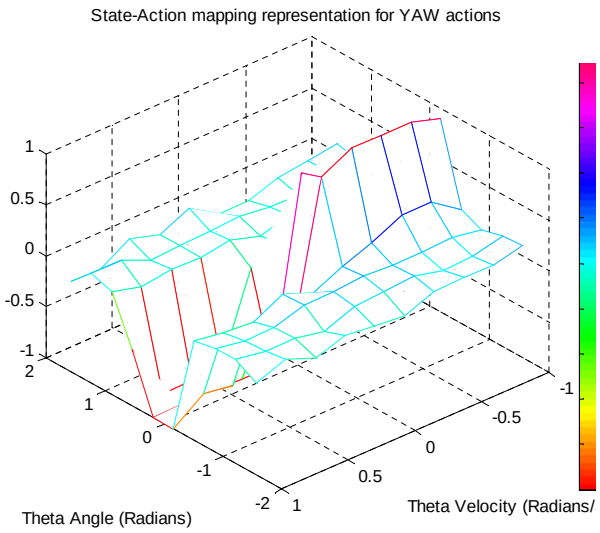


Figure 9. Theta angle - Theta velocity mapping of a trained rob
controller. Colorbar on the right represents the actions taken.

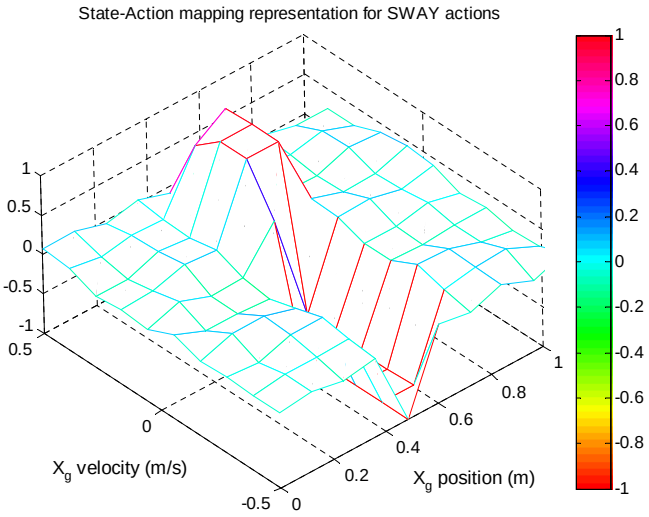


Figure 10. Centroid X position - Centroid X velocity mapping of a trained robot controller. Colorbar on the right represents the actions taken.

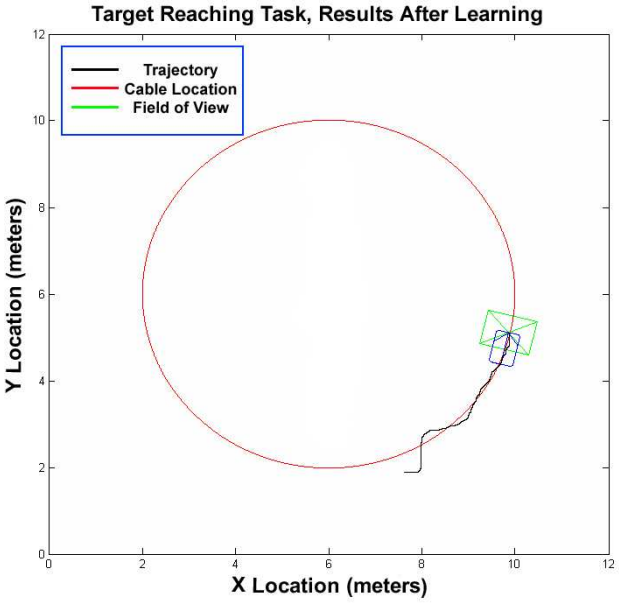


Figure 11. Behavior of a trained robot controller, results of the simulated cable tracking task after learning period is completed.

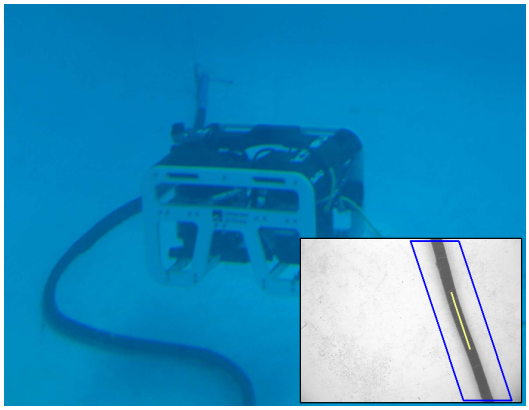


Figure 12. $ICTINEU^{AUV}$ in the test pool. Small bottom-right image: Detected cable.

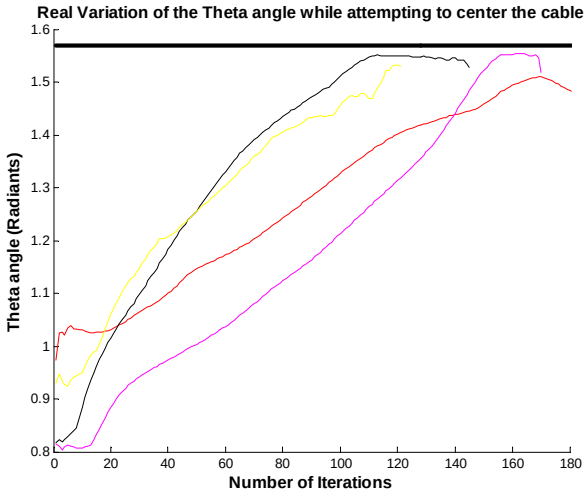


Figure 13. 4 real measured trajectories of the θ angle of the image plane while attempting to center the cable at $\pi/2$ rad. Every iteration represents 0.1 seconds.

jectories and other controllers.

ACKNOWLEDGEMENTS

First of all, we would like to give our special thanks to the group of the University of the Balearic Islands for allowing us to use their cable detection algorithm. This work has been financed by the Spanish Government Comission MCYT, project number DPI2005-09001-C03-01, also partially funded by the MOMARNET EU project MRTN-CT-2004-505026 and the European Research Training Network on Key Technologies for Intervention Autonomous Underwater Vehicles FREESUBNET, contract number MRTN-CT-2006-036186.

REFERENCES

[1] D. A. Aberdeen, *Policy-Gradient Algorithms for Partially Observable Markov Decision Processes*, Ph.D. dissertation, Australian National University, April 2003.

[2] C. Anderson, 'Approximating a policy can be easier than approximating a value function', Computer science technical report, University of Colorado State, (2000).

[3] J. Antich and A. Ortiz, 'Underwater cable tracking by visual feedback', in *First Iberian Conference on Pattern recognition and Image Analysis (IbPRIA, LNCS 2652)*, Port d'Andratx, Spain, (2003).

[4] C.G. Atkenson, A.W. Moore, and S. Schaal, 'Locally weighted learning', *Artificial Intelligence Review*, **11**, 11–73, (1997).

[5] J.A. Bagnell and J.G. Schneider, 'Autonomous helicopter control using reinforcement learning policy search methods', in *Proceedings of the IEEE International Conference on Robotics and Automation*, Korea, (2001).

[6] J. Baxter and P. Bartlett, 'Direct gradient-based reinforcement learning: I. gradient estimation algorithms', Technical report, Australian National University, (1999).

[7] A. El-Fakdi, M. Carreras, and P. Ridao, 'Towards direct policy search reinforcement learning for robot control', in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (2006).

[8] T. I. Fossen, *Guidance and Control of Ocean Vehicles*, John Wiley and Sons, 1995.

[9] Bradley Hammer, Sanjiv Singh, and Sebastian Scherer, 'Learning obstacle avoidance parameters from operator behavior', *Journal of Field Robotics, Special Issue on Machine Learning Based Robotics in Unstructured Environments*, **23 (11/12)**, (December 2006).

[10] N. Kohl and P. Stone, 'Policy gradient reinforcement learning for fast quadrupedal locomotion', in *IEEE International Conference on Robotics and Automation (ICRA)*, (2004).

[11] V.R. Konda and J.N. Tsitsiklis, 'On actor-critic algorithms', *SIAM Journal on Control and Optimization*, **42, number 4**, 1143–1166, (2003).

[12] L.J. Lin, 'Self-improving reactive agents based on reinforcement learning, planning and teaching.', *Machine Learning*, **8(3/4)**, 293–321, (1992).

[13] P. Marbach and J. N. Tsitsiklis, 'Gradient-based optimization of Markov reward processes: Practical variants', Technical report, Center for Communications Systems Research, University of Cambridge, (March 2000).

[14] T. Matsubara, J. Morimoto, J. Nakanishi, M. Sato, and K. Doya, 'Learning sensory feedback to CPG with policy gradient for biped locomotion', in *Proceedings of the International Conference on Robotics and Automation ICRA*, Barcelona, Spain, (April 2005).

[15] N. Meuleau, L. Peshkin, and K. Kim, 'Exploration in gradient based reinforcement learning', Technical report, Massachusetts Institute of Technology, AI Memo 2001-003, (April 2001).

[16] R. Munos and A. Moore, 'Barycentric interpolators for continuous space and time reinforcement learning', (1998).

[17] A. Ortiz, M. Simo, and G. Oliver, 'A vision system for an underwater cable tracker', *International Journal of Machine Vision and Applications*, **13 (3)**, 129–140, (2002).

[18] J. Peters and S. Schaal, 'Policy gradient methods for robotics', in *IEEE/RSJ International Conference on Intelligent Robots and Systems IROS'06*, Beijing, China, (October 9-15 2006).

[19] D. Ribas, N. Palomeras, P. Ridao, M. Carreras, and E. Hernandez, 'Ictineu auv wins the first sauc-e competition', in *IEEE International Conference on Robotics and Automation*, (2007).

[20] P. Ridao, A. Tiano, A. El-Fakdi, M. Carreras, and A. Zirilli, 'On the identification of non-linear models of unmanned underwater vehicles', *Control Engineering Practice*, **12**, 1483–1499, (2004).
[21] M.T. Rosenstein and A.G. Barto, 'Robot weightlifting by direct policy search', in *Proceedings of the International Joint Conference on Artificial Intelligence*, (2001).
[22] W.D. Smart, *Making Reinforcement Learning Work on Real Robots*, Ph.D. dissertation, Department of Computer Science at Brown University, Rhode Island, May 2002.
[23] R. Sutton and A. Barto, *Reinforcement Learning, an introduction*, MIT Press, 1998.
[24] R.S. Sutton, D. McAllester, S. Singh, and Y. Mansour, 'Policy gradient methods for reinforcement learning with function approximation', *Advances in Neural Information Processing Systems*, **12**, 1057–1063, (2000).
[25] R. Tedrake, T. W. Zhang, and H. S. Seung, 'Stochastic policy gradient reinforcement learning on a simple 3D biped', in *IEEE/RSJ International Conference on Intelligent Robots and Systems IROS'04*, Sendai, Japan, (September 28 - October 2 2004).