An Hybrid Methodology for RL-based Behavior Coordination in a Target Following Mission with an AUV

M. Carreras[1], J. Yuh[2] and J. Batlle[1]

[1] Institute of Informatics and Automation
University of Girona
Edifici Politècnica II, Campus Montilivi,
17071 Girona, Spain

[2] Autonomous Systems Laboratory
University of Hawaii
2540 Dole St., Holmes 302
Honolulu, HI 96822, USA

Abstract - This paper proposes a behavior-based scheme for high-level control of Autonomous Underwater Vehicles (AUVs). Two main characteristics can be highlighted in the control scheme. Behavior coordination is done through a hybrid methodology, which takes in advantages of the robustness and modularity in competitive approaches, as well as optimized trajectories in cooperative ones. As a second feature, behavior state/action mapping is learnt by means of Reinforcement Learning (RL). A continuous Q-learning algorithm, implemented with a feed-forward neural network, is used. The behavior-based scheme attempts to fulfill simple missions in which several behaviors/tasks compete for the vehicle's control. This paper shows its feasibility with a target following mission designed to be carried out in a pool with the AUV ODIN. In this paper, simulation results are shown demonstrating the good performance of the hybrid method on behavior coordination as well as the convergence of the RL-based behaviors.

I. INTRODUCTION

The control of an autonomous vehicle to fulfill a mission in an unstructured and unknown environment is still a challenge. In the middle of 1980s the appearance of *Behavior-based Robotics* [1] philosophy revolutionized the development of robots. Its principles of parallelism, modularity, situatedness/embeddedness and behavior emergence provided a more feasible approach than traditional top-down *deliberative* architectures. Behavior-based control architectures propose a bottom-up methodology in which several behaviors or tasks act independently generating the set-points to be followed by the robot. Behaviors are implemented as a control law using inputs and outputs. A *coordination* module is in charge of choosing the final set-point to be followed. Two main coordination methodologies can be found. In *competitive* coordinators a single behavior is selected whereas in *cooperative* coordinators several behavior responses are superposed.

According to the coordination system, some advantages and disadvantages appear in the control performance of an autonomous vehicle. After testing 4 well-known behavior-based architectures (Subsumption [2], Action Selection Dynamics [3], Schema-based approach [4] and Process Description Language [5]) in a simulated 3D-navigation mission with an AUV some conclusions were extracted [6,7]. Competitive methods (subsumption and action selection dynamics) show good robustness in the behavior selection and modularity when adding new behaviors. However, a non-optimized trajectory is found when there is a continuous change of the dominant behaviors. As far as cooperative methods are concerned, they have an optimal trajectory when parameters are properly tuned. However, they lack of robustness. A small change on the parameters can lead to control failures. In some circumstances, a set of behaviors can cancel the action of behaviors with a higher priority (i.e. obstacle avoidance behaviors).

In the implementation of a behavior-based system, the design and tune-up of the behaviors is a hard task and requires a lot of experimentation. In these systems, there is also the need of performing in unknown and time-varying environments, which means that some kind of adaptation is needed. To solve these difficulties, many robotic systems include learning techniques. There is not yet an established methodology to develop adaptive behavior-based systems. However, a commonly used approach is *Reinforcement Learning* (RL) [8], a class of learning algorithm in which an agent tries to maximize a scalar evaluation (reward or punishment) of its interaction with the environment. A RL system tries to map the states of the environment to actions (policy) in order to obtain the maximum reward. Most RL techniques are based on F*inite Markov Decision Processes* (FMDP) causing that state and action spaces are finite. The main advantage of RL is that it does not use any knowledge database, as in most forms of machine learning, making this class of learning suitable for online learning. The main disadvantages are the large convergence time and the lack of generalization among continuous variables, which represent one of the current research topics in RL.

RL has been applied to various behavior-based systems, most of them using Q_learning [9]. In some cases, the RL algorithm was used to adapt the coordination system [10, 11]. On the other hand, some researchers have used RL to learn the internal structure of the behaviors, mapping the perceived states to robot actions [12, 13, 14]. The work presented by Mahadevan [12] demonstrated that the decomposition of the whole agent learning policy in a set of behaviors, as Behavior-based robotics proposes, simplified and increased the learning speed.

This paper presents a new behavior-based high-level control approach for AUVs. The new approach proposes a hybrid coordination method between competitive and cooperative ones. The coordination depends on a hierarchy of behaviors. Depending on an activation level associated to each behavior response, the coordinator superposes the responses in a different grade. The final behavior of the vehicle has the robustness of competitive methods, when a priority behavior becomes completely active, and the optimized trajectories of cooperative methods, in the other situations. As a second important feature, the behavior-based scheme uses RL algorithms to learn the internal mapping between states and actions of each behavior. In the work presented in this paper, a continuous implementation of the Q_learning algorithm was used. Generalization between states and actions was achieved by a feed-forward neural network which approximates the Q_function. Direct Q_learning [15] (backpropagation) was used to train the network.

To test the feasibility of the hybrid coordination method and RL-based behaviors, a target following mission in a pool environment was designed for the AUV ODIN [16]. Two behaviors are in charge of carrying out the mission. The "obstacle avoidance" behavior uses the eight sonar sensors of the vehicle to stay away from objects, walls, etc. The "target following" behavior uses a video camera to detect the relative target position. The target is detected by color segmentation. The paper shows simulation results of the mission achievement. Convergence of the RL-based behaviors is shown as well as efficiency and robustness of the hybrid coordinator. The hydrodynamics model of ODIN [17], its control system [18] and a virtual pool were used in the simulations.

The structure of this paper is as follows. Section 2 describes the proposed hybrid coordination system. Section 3 introduces the continuous Q_learning algorithm used for behavior learning. In section 4, the target following application to test the hybrid coordination method is detailed. In section 5, simulation results are given. And finally, conclusions and future work are presented in section 6.

II. HYBRID COORDINATOR

Minimizing disadvantages and maximizing advantages of competitive and cooperative methodologies, a *hybrid coordination* method is proposed. In the proposed method, the coordination of the responses is done through a hybrid approach that keeps the robustness and modularity of competitive approaches as well as the optimized paths of cooperative ones.

The coordinator is based on normalized behavior outputs. The outputs contain a three-dimensional vector "v_i" which represents the velocity proposed by the behavior. Associated with this vector is an activation level "a_i" which indicates how important it is for the behavior to take control of the robot. This value is between 0 and 1, see Fig. 1. This codification sharply defines the control action from the activation of the behavior.

The proposed coordination system is composed of a set of *hierarchical hybrid nodes,* see Fig. 2. The nodes have two inputs and generate a merged normalized control response. The nodes compose a hierarchical and cooperative coordination system. The idea is to use the optimized paths of cooperation when the predominant behavior is not completely active. The nodes have a dominant behavior which suppresses the responses of the non-dominant behavior when the first one is completely activated (a_i=1). However, when the dominant behavior is partially activated ($0<a_i<1$), the final response will be a combination of both inputs. Non-dominant behaviors can slightly modify the responses of dominant behaviors when they aren't completely activated. For example, if the dominant behavior is "obstacle avoidance" and the non-dominant is "go to point", when "obstacle avoidance" is only slightly activated (the obstacles are still far), a mixed response will be obtained. When non-decisive situations occur, cooperation between behaviors is allowed. Nevertheless, robustness is present when dealing with critical situations.

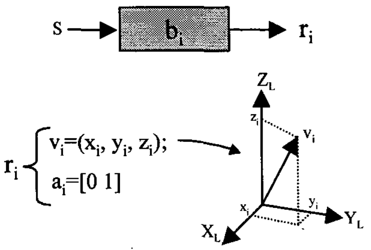


Fig. 1. Normalized output of a behavior.

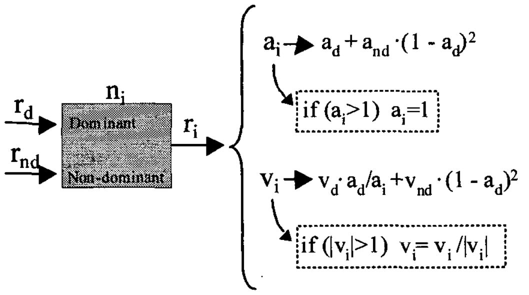

n_i
r_d
r_{nd}
Dominant
Non-dominant
r_i
$a_i \to a_d + a_{nd} \cdot (1 - a_d)^2$
if (a_i>1) a_i=1
$v_i \to v_d \cdot a_d/a_i + v_{nd} \cdot (1 - a_d)^2$
if ($|v_i|$>1) $v_i = v_i / |v_i|$

Fig. 2. Hierarchical hybrid node.

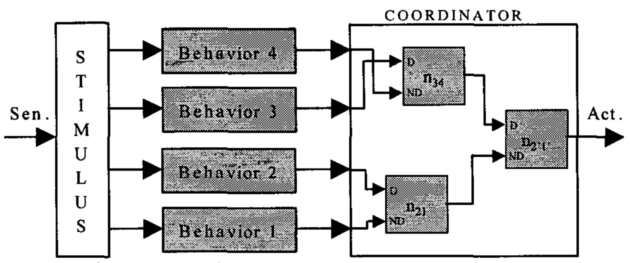


Fig. 3. Behavior-based architecture with the hybrid coordination system.

The node "n_i" has the ability to generate a normalized response like the one generated by behaviors. The effect of the non-dominant behavior depends on the squared activation of the dominant to assure that in a critical situation between both, the dominant will always take control. Depending on the situation, the control response could be produced by all the behaviors or by only one. The hybrid nodes do not need a tuning phase. The coordination of a set of behaviors is defined hierarchically classifying each behavior depending on its priority. A disposition of the whole coordination system using hierarchical hybrid nodes can be seen in Fig. 3.

The coordination method can be classified as a *hybrid* approach because the response is the one generated by the dominant behavior affected by non-dominant behaviors according to the level of activation of the first. Although to the authors best knowledge there is no hybrid coordination system presented in the literature, this method offers good properties and can be successfully implemented in an autonomous robot. The proposed method has been implemented in simulation [7] showing its optimized paths, robustness and modularity controlling an AUV in a 3D-navigation mission.

III. RL-BASED BEHAVIORS

Reinforcement Learning (RL) [8] is a class of learning suitable for robots when online learning without information about the environment is required. In RL an agent tries to maximize a scalar evaluation (reward or punishment) of its interaction with the environment. The evaluation is generated by the *critic* using an utility function. A RL system tries to map the states of the environment to actions (policy) in order to obtain the maximum reward. In our case, the state is the sensor information perceived by the robot and the action is the behavior output (the velocity set-points). RL does not use any knowledge database as in most forms of machine learning. Most theories are based on Finite Markov Decision Processes (FMDPs).

The approach taken in this paper was a continuous implementation of the Q_learning algorithm [9]. Q_learning is a temporal difference [8] algorithm, which means that the transition probabilities between the states of the FMDPs are not required, and therefore, the dynamics of the environment does not have to be known. Temporal difference methods are also suitable to learn in an incremental way, required in online robot learning. An important characteristic of Q_learning is that is an off-policy algorithm. The optimal action values are leant independently of the policy being followed. This is very important in our behavior-based architecture because all the behaviors can be learnt even if they are not controlling the vehicle.

The original Q_learning algorithm is based on FMDPs. It uses the states perceived (*s*), the actions taken (*a*) and the reinforcements received (*r*) to update the values of a table, denoted as $Q(s,a)$. If state/action pairs are continually visited, the Q values converge to a *greedy policy*, in which the maximum Q value for a given state points to the optimal action. Fig. 4 shows the Q_learning algorithm.

There are several parameters which define the learning evolution:

- γ: discount rate [0 1]. Concerning the maximization of future rewards. If γ=0, the agent is "myopic" in being concerned only with maximizing immediate rewards.
- α: learning rate [0 1].
- $\in$: random action probability [0 1]. The agent needs to explore new actions in order to find the optimal ones, but also needs to exploit the best actions to accumulate the maximum reward, (Exploitation / Exploration dilemma). The final action is called $\in$-*greedy* action.

```
1. Initialize $\hat{Q}(s,a)$ arbitrarily
2. Repeat:
   (a) $s_t \leftarrow$ the current state
   (b) choose an action $a_t$ that maximizes $\hat{Q}(s_t,a)$ over all $a$
   (c) $\varepsilon$-greedy action, carry out action $a_t$ in the world with probability
       $(1-\varepsilon)$, otherwise apply a random action (exploration)
   (d) Let the short term reward be $r_t$, and the new state be $s_{t+1}$
   (e) $\hat{Q}(s_t,a_t) = \hat{Q}(s_t,a_t) + \alpha[r_t + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1},a_{t+1}) - \hat{Q}(s_t,a_t)]$
```

Fig. 4. Q_learning algorithm.

Due to its use of finite spaces, Q-learning has a considerably large learning time and memory requirement. More sophisticated methods [19,20] implement a parameterized Q-function which enables generalization between states and actions. In this paper a continuous implementation of the algorithm was used. A neural network approximates the Q_function and its weights are updated according to the backpropagation algorithm [21], also known as *direct Q_learning* [15]. There is no convergence proof of this continuous implementation. However, with suitable network configuration and parameter selection, the algorithm demonstrated to converge. The *neural Q_learning* algorithm structure is showed in Fig. 5. The neural network approximates the Q_function:

$$\hat{Q}(s_t,a_t) = r_t + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1},a_{t+1})$$

therefore, its inputs are the continuous states and actions, and the output is the Q_value. According to the output value, the error is found and the weights are updated using the standard backpropagation algorithm. Sigmoidal and lineal activation functions are used. Initially the weights are generated randomly inside a range that produces an activation level between the lineal and non-lineal zone of the sigmoidal functions. To find the action that maximizes the Q_value, the network evaluates all the possible actions that could be applied. Although actions are continuous, a finite set, which guarantee enough resolution, is used.

In order that each behavior learns to act as it is expected, a reinforcement function must be defined, see Fig. 5. This function associates each state with a reward "*r*" (-1, 0 and 1). By associating the desired states with "*r=1*" and the undesired with "r=-1", the algorithm learns how to act.

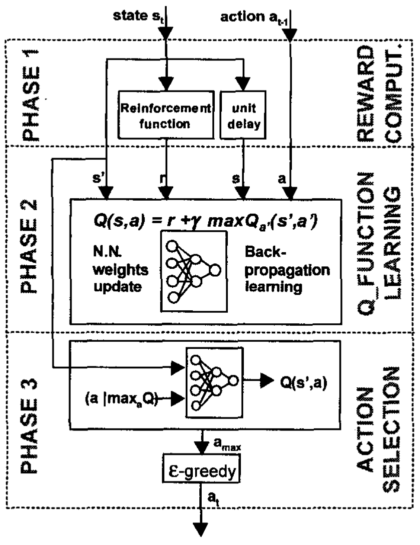


Fig. 5. Neural Q_learning algorithm structure.

IV. TARGET FOLLOWING MISSION

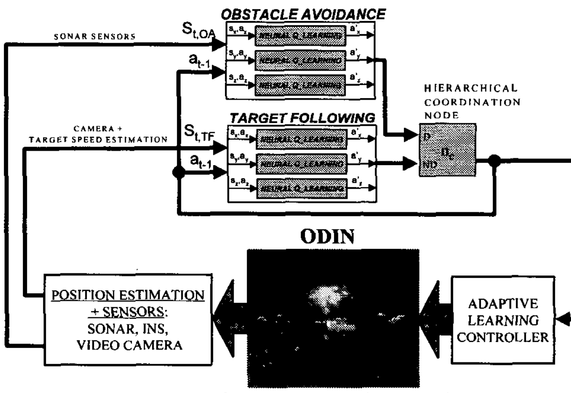
To test the feasibility of the hybrid coordinator and RL-based behaviors, a target following application was designed. The proposed mission consists of following a target by means of a camera and avoiding obstacles using a set of sonar sensors. The AUV must act as an autonomous camera recording all the movements of the target without colliding or losing the target. This application was designed to be carried out in a swimming pool where light absorption does not apply.

The vehicle for which the mission was conceived is ODIN [16] (Underwater Robotic Intelligent System), the testbed AUV developed at the Autonomous Systems Laboratory of the University of Hawaii. ODIN is a sphere shaped vehicle with eight thrusters (4 horizontal and 4 vertical). It is capable of maneuvering with six degrees-of-freedom (DOF). ODIN has various navigation sensors such as 8 sonar transducers, a pressure sensor, and an inertial navigation system; and an on-board CPU with VxWorks OS in VMEbus. In this paper, the mission is fulfilled using simulations with the ODIN's dynamics model [17], the adaptive learning controller [18] and a virtual pool as environment. Further work will be based on real experiments.

To accomplish this mission a Behavior-based architecture with two behaviors was designed. Each behavior receives information about the current state, usually from sensors, and it also receives the last action taken. This action is used by the neural_q_learning algorithm to update the neural network weights. Each behavior generates a 3D-speed vector and an activation level, which determines, according to the hierarchy of behaviors, the final output. Once the coordination phase is done, the output is sent to the adaptive learning controller. Fig. 6 shows the schema of the architecture. The two behaviors are:

- *Obstacle avoidance.* The goal is to avoid any obstacles perceived by means of 8 sonar sensors, see Fig. 7. It has more priority than the *target following* behavior. The behavior is learnt using the neural Q_learning algorithm for each DOF. The gravity center of the 8 sonar values is found and each relative coordinate (x,y,z) is used as input. A reinforcement function gives different rewards (-1, 0 or 1) depending on the distance at which obstacles are detected. The activation level is proportional to the proximity of obstacles. When obstacles are very close, the activation level is 1 and the behavior has the whole control on the vehicle.

- *Target following.* The goal of this behavior is to follow a target using a video camera pointed towards X-axis, see Fig. 7. As the mission was designed to be carried out in a swimming pool, a simple segmentation algorithm is used to detect the position of a tinted target. The behavior is also learnt using the neural Q_learning algorithm for each DOF (x,y,z). The inputs of each DOF are the relative position of the target and an estimation of the target speed. The latter one is needed to be able to follow moving targets. A reinforcement function gives positive rewards ($r=1$) if the target is around the vehicle's relative position x=5, y=0 and z=0. Otherwise, values like $r=0$ or $r=-1$ are given. The activation level is 1 when the target is detected, alternatively, it is 0.



OBSTACLE AVOIDANCE
SONAR SENSORS
$S_{t,OA}$
a_{t-1}
NEURAL Q_LEARNING
HIERARCHICAL COORDINATION NODE
TARGET FOLLOWING
CAMERA + TARGET SPEED ESTIMATION
$S_{t,TF}$
ODIN
POSITION ESTIMATION + SENSORS: SONAR, INS, VIDEO CAMERA
ADAPTIVE LEARNING CONTROLLER

Fig. 6. Target following mission architecture.

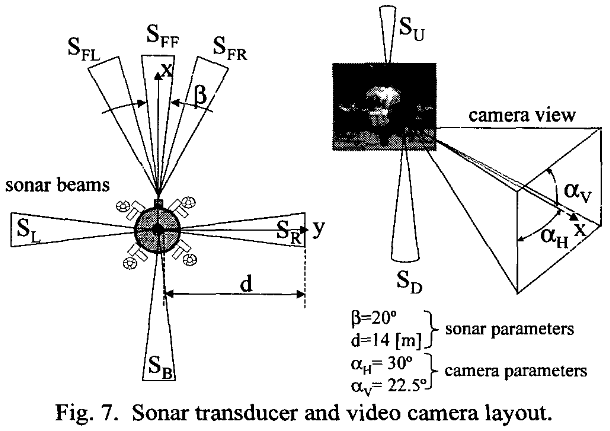


Fig. 7. Sonar transducer and video camera layout.

V. SIMULATION RESULTS

The architecture proposed above was implemented in simulation with a virtual underwater environment, see Fig. 8. The ODIN dynamics model [17] and the adaptive learning controller [18] were used. A moving target was introduced carrying out a 3D closed path repeatedly. The velocity of the target changed between 0 and 0.2 m/s (60% of the maximum velocity of ODIN).

"Target following" and "Obstacle avoidance" behaviors were implemented using the neural Q_learning algorithm. Many simulation episodes were done in order to find the optimal neural network configuration. Finally a 3 layer neural network was used. The parameters and specifications of the "*obstacle avoidance*" and "*target following*" behaviors can be seen in tables I and II respectively.

Each degree of freedom was implemented independently with its inputs/outputs and rewards. At the beginning, each behavior was learnt alone, without the influence of the other. The number of iterations required to learn each DOF was approximately 2000 (sample time=1s).

Fig. 9 shows the evolution of the "x" DOF of the "target following" behavior during its training. It can be seen how the algorithm explores the action space and learns how to track the target. Fig. 10 shows the state/action mapping of the same behavior after the training. And Fig. 11 shows the accumulated rewards obtained in 13 different learning trials of the same behavior, during 10 episodes of 1000 iterations. The rewards are accumulated over one episode, which means that the maximum value (if the vehicle follows the target perfectly) is 1000. It can be seen that during the first trials the accumulated rewards fluctuated. However, in most of the trials, after the fifth episode, the accumulated reward was bigger than 900, and in some of them was 1000. This convergence proof can also be seen in the average line, which continually increased.

Once the 3 DOFs of both behaviors were completely learnt, the mission was tested. Fig. 12 shows the tracking error evolution during the mission. When the vehicle was close to an obstacle, the obstacle avoidance behavior took partial control of the vehicle, and therefore, the tracking error increased. However, the hybrid coordination system generated a cooperative response between both behaviors, and the target was not lost.

In Fig. 13, the three-dimensional path of the target and the ODIN vehicle can be seen. The vehicle, pointing to the pool y-axis, follows the target at 5 meters distance. The figure demonstrates the good performance of the behavior-based architecture in accomplishing the mission. In Fig. 14 and 15 a top and lateral views of the same path are shown. In these views it can be appreciated that the vehicle stops in following the target on the right (Fig. 14) and on the bottom (Fig.15), due to the presence of the wall and the bed of the pool respectively.

TABLE I
OBSTACLE AVOIDANCE BEHAVIOR SPECIFICATIONS.

OBSTACLE AVOIDANCE BEHAVIOR	
Input variables	8 sonar transducers values
Codification	x_o,y_o,z_o (gravity center of the perceived obstacles, [-4 4] m)
Output variables	a_x ,a_y ,a_z (desired vehicle speed in x,y,z [0 1])

Critic function	If dist > 3.2 m : $r_t = 1$
	else if dist > 1.6 m : $r_t = 0$
	else $r_t = -1$ (dist = abs(x_o) or abs(y_o) or abs(z_o))

Behavior activation	act=dist/3 (if act>1, act =1)
Q_learning param.	$\alpha = 0.1$; $\gamma = 0.9$; $\epsilon = 0.2$
Neural Network	*inputs : 2 (i.e.: x_o a_x) outputs : 1 (Q_val)* *layers*: 1- 5 neurons; sigmoidal act. function 2- 3 neurons; sigmoidal act. function 3- 1 neuron; lineal act. function

TABLE II
TARGET FOLLOWING BEHAVIOR SPECIFICATIONS

TARGET FOLLOWING BEHAVIOR	
Input variables	- x_t,y_t,z_t (errors between the target position and the target desired location, [-3 3]m) - v_{xt},v_{yt},v_{zt} (target velocity estim. [-0.3 0.3] m/s)
Output variables	- a_x ,a_y ,a_z (desired vehicle speed in x,y,z [0 1])

Critic function	If dist > 2 m : $r_t = -1$
	else if dist > 0.5 m : $r_t = 0$
	else $r_t = 1$ (dist = abs(x_t) or abs(y_t) or abs(z_t))

Behavior activation	act=1 if the target is visible, alternatively 0.
Q_learning param.	$\alpha = 0.1$; $\gamma = 0.9$; $\epsilon = 0.2$
Neural Network	*inputs : 3 (i.e.: x_t v_{xt} a_x) outputs : 1 (Q_val)* *layers:* 1- 4 neurons; sigmoidal act. function 2- 2 neurons; sigmoidal act. function 3- 1 neurons; lineal act. function

VI. CONCLUSIONS

This paper has proposed a behavior-based scheme for high-level control of Autonomous Underwater Vehicles (AUVs). The scheme was compound by a hybrid coordination system and several Reinforcement Learning-based behaviors. The method has been tested in a simulated target following experiment with the AUV ODIN. The behaviors have been implemented using a continuous implementation of the Q-learning algorithm. The simulation results showed the feasibility of the hybrid approach as well as the convergence of the learning algorithms. The proposed hybrid coordination demonstrated as behaving with the robustness of competitive coordinators and with the optimized paths of cooperative ones. The neural network implementation of the Q_learning algorithm also demonstrated to converge to the optimal policy, obtaining the maximum accumulated rewards.

Future work will concentrate on the realization of real experiments and on the improvement of the RL-based behaviors in order to learn simultaneously all the behaviors and to use only one RL function for each behavior.

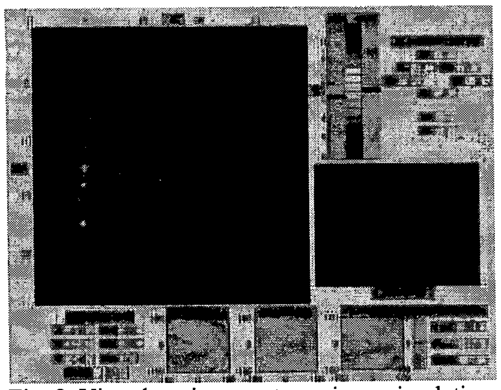


Fig. 8, Virtual environment running a simulation.

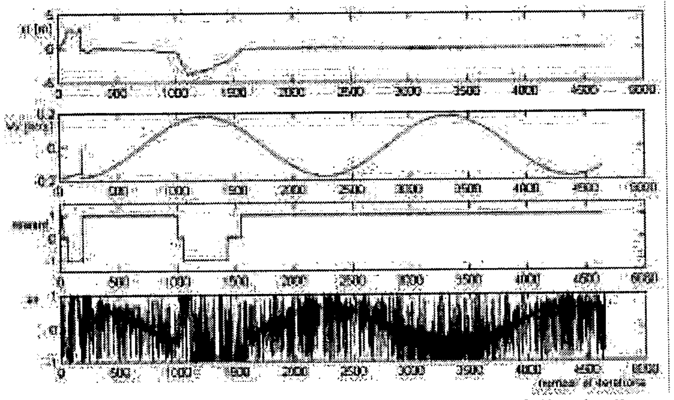


Fig. 9. Typical learning evolution of the "target following" behavior in x axis. The state (Xt, Vx), the reward and the action a_x signals can be shown.

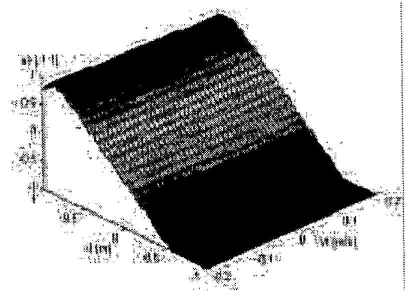


Fig. 10. State/action mapping of the " target following" in x.

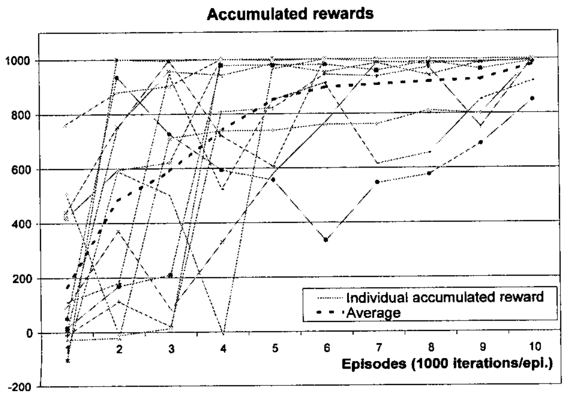


Fig. 11. Convergence of the "target following" in x axis.

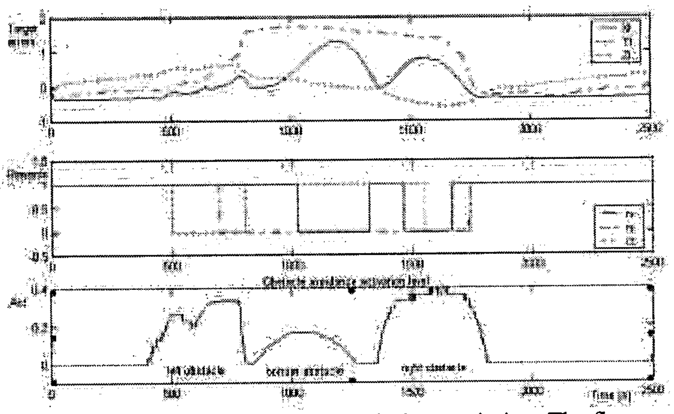


Fig. 12. Tracking error evolution during a mission. The figure shows the error and rewards in x,y,z, and the activation level of the obstacle avoidance behavior.

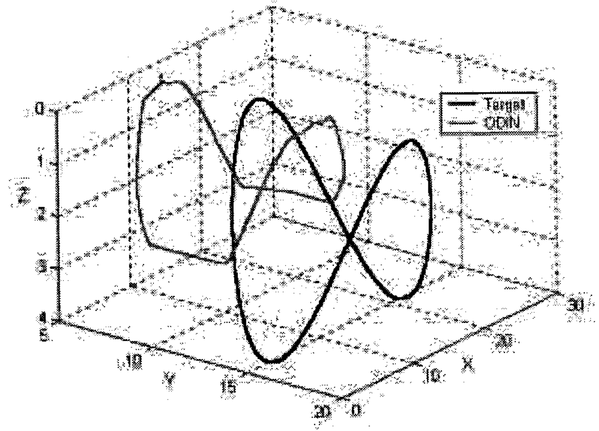


Fig. 13. Target and ODIN 3D paths after the mission.

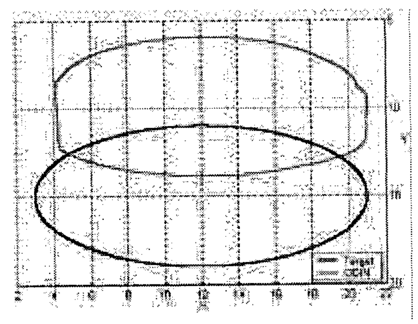


Fig. 14. Target and ODIN paths, top view.

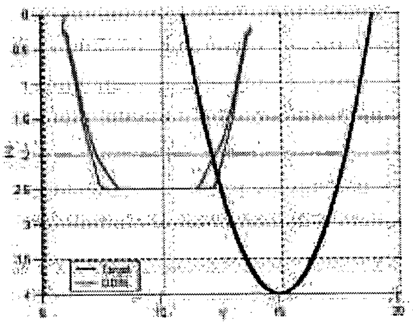


Fig. 15. Target and ODIN paths, lateral view.

REFERENCES

[1] Arkin, R. Behavior-based Robotics. MIT Press, 1998.

[2] Brooks, R. A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, vol. RA-2, is.1, pp.14-23. 1986.

[3] Maes, P. Situated Agents Can Have Goals. *Robotics and Automation Systems*, vol. 6, pp. 49-70, 1990.

[4] Arkin, R. C. Motor schema-based mobile robot navigation. *International Journal of Robotica Research*, vol. 8, is. 4, pp. 92-112, 1989.

[5] Steels, L. Building agents with autonomous behaviour systems. The artificial route to artificial intelligence. *Building situated embodied agents*. Lawrence Erlbaum Associates, New Haven, 1993.

[6] Carreras, M., Batlle, J., Ridao, P. and Roberts, G.N.. An overview on behaviour-based methods for AUV control. *MCMC2000, 5th IFAC Conference on Manoeuvring and Control of Marine Crafts.* Aalborg, Denmark, August 2000.

[7] Carreras, M.. An Overview of Behaviour-based Robotics with simulated implementations on an Underwater Vehicle. *University of Girona, Spain. Informatics and Applications Institute.* Research report: IIiA 00-14-RR. October, 2000.

[8] Sutton, R. and Barto, A. *Reinforcement Learning, an introduction*. MIT Press, 1998.

[9] Watkins, C.J.C.H., and Dayan, P. Q-learning. *Machine Learning*, 8:279-292, 1992.

[10] Maes, P. and Brooks, R. Learning to coordinate behaviors. In *Proceedings of the Eighth AAAI*, pages 796-802. Morgan Kaufmann, 1990.

[11] Gachet, D., Salichs, M., Moreno, L. and Pimental, J. Learning Emergent tasks for an Autonomous Mobile Robot, *Proceedings of the International Conference on Intelligent Robots and Systems (IROS '94)*, Munich, Germany, September, pp. 290-97, 1994.

[12] Mahadevan, S. and Connell, J. Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55:311-365, 1992.

[13] Shackleton, J. and Gini, M. Measuring the Effectiveness of Reinforcement Learning for Behavior-based Robots. *Adaptive Behavior*, 1997.

[14] Touzet, C. Neural reinforcement learning for behaviour synthesis. In: *Robotics and Autonomous Systems*, 22, 251-281, 1997.

[15] Baird, K. Residual Algorithms: Reinforcement Learning with Function Approximation. *Machine Learning: Proceedings of the Twelfth International Conference*, San Francisco, USA, 1995.

[16] Choi, S.K., Yuh, J., Takashige, G.Y.. Development of the Omni-Directional Intelligent Navigator. *IEEE Robotics and Automation Magazine*, pp. 44-53, 1995.

[17] Nie, J., Yuh, J., Kardash, E., and Fossen, T.I.. Onboard sensor-based adaptive control of small UUVs in the very shallow water. *Proc. of IFAC-Control applications in Marine Systems*, Fukuoka, Japan, pp. 201-206, 1998.

[18] J. Yuh, An adaptive and learning control system for underwater robots, Proc. 13th World Congress IFAC, San Francisco, CA, June 1996, Vol. A, pp. 145-150.

[19] Gaskett, C., Wettergreen, D. and Zelinsky, A. Q-learning in continuous state and action spaces. In *Proc. of the 12th Australian Joint Conference on Artificial Intelligence*, Sydney, Australia, 1999.

[20] Takahashi, Y., Takada, M. and Asada, M. Continuous Valued Q-learning for Vision-Guided Behavior Acquisition. In *International Conference on Multisenso Fusion and Integration for Intelligent Systems*, pages 716-721, 1999.

[21] Haykin, S. Neural Networks, a comprehensive foundation. Prentice Hall, 2nd ed., 1999.