24th Mediterranean Conference on Control and Automation (MED)
June 21-24, 2016, Athens, Greece

Reinforcement Learning-based Motion Planning of a Triangular Floating Platform under Environmental Disturbances

Konstantinos Tziortziotis, Kostas Vlachos, *Member, IEEE*, and Konstantinos Blekas, *Member, IEEE*

***Abstract*— This paper investigates the use of reinforcement learning for the motion planing of an autonomous triangular marine platform in unknown environments under various environmental disturbances. The marine platform is over-actuated, i.e. it has more control inputs than degrees of freedom. The proposed approach uses an online least-squared policy iteration scheme for value function approximation in order to estimate optimal policy. We evaluate our approach in simulation, taking under consideration the dynamics of the platform, the dynamics and limitations of the actuators, under the presence of wind, and sea current disturbances. We report simulation results concerning its performance on estimating optimal navigation policies to unknown environments. Despite the model dynamics, the actuation dynamics and limitations, and the environmental disturbances, the presented results are promising.**

I. Introduction

Marine vehicle navigation aims at finding a route through obstacles and constructing a motion planner in terms of a feasible sequence of actions that allow to move a marine vehicle from an initial "configuration" to a goal "configuration". Ideally, such planner tries to optimize an objective function consisting of attributes such as plan duration, energy expanded, etc.

There is a tremendous need for developing fast analytic algorithms for predicting the collision probability due to model uncertainty and random disturbances in the environment for a planar holonomic vehicle such as a marine surface vessel [1], [2]. These predictions lead to a robust motion planning algorithm that discovers the optimal motion plan quickly and efficiently. Incorporating model learning into the predictions exhibits emergent active learning strategies to safely and effectively complete the mission.

A flexible framework for motion planning and autonomous vehicle navigation is through Reinforcement Learning (RL) [3], [4]. RL aims at controlling an autonomous agent in unknown stochastic environments. Typically, the environment is modeled as a Markov Decision Process (MDP), where the agent receives a scalar reward signal that evaluates every transition. The objective is to maximize its long-term profit that is equivalent to maximizing the expected total discounted reward. Value function is used for measuring the quality of a policy, which associates to every state the expected discounted reward when starting from this state and all decisions are made following the particular policy. A plethora of methods have been proposed during the last two decades using a variety of value-function estimation techniques [4], [5].

The temporal difference algorithms provide a nice framework for policy evaluation since they have the flexibility to handle large or continuous state space of real world applications. More specifically, least-squares temporal difference (LSTD) family of methods is very popular mechanism for approximating the value function that performs an iterative procedure for optimal policy estimation. Finally, *model-based* approaches for value function approximation have been also proposed based on on-line schemes, through Gaussian processes [6], clustering schemes [7], or regression tree models [8], [9].

In the literature there are some marine robotic applications, mostly involving autonomous underwater vehicles (AUV), using reinforcement learning, see for a survey in [10], and in [20]. In [10] for example a neural networks-based reinforcement learning scheme is presented for high-level control of AUV's. In [11] another approach is proposed for motion planning of under-actuated AUV in unknown non-uniform sea flow. A recent work, presented in [12], proposes a path planning algorithm for an under-actuated marine vehicle in the presence of ocean current disturbances, based on reinforcement learning. Nevertheless, the marine vehicle is described only by the kinematic model.

Our work focuses on the development of an intelligent navigation mechanism based on reinforcement learning, for the over-actuated autonomous triangular marine platform shown in Fig. 1. To our knowledge, this is the first time that a reinforcement learning scheme is proposed for the autonomous navigation of an over-actuated marine vehicle. The detailed model of the platform can be found in [13]. Controllers for the problem of the autonomous dynamic positioning of the platform have been proposed in [13] and [14]. Here, we examine the problem of the determination of a desired path in an unknown environment under various environmental disturbances. The required forces and moment are provided by three rotating pump jets, consequently the system is over-actuated, i.e., it has more control inputs than degrees of freedom (DOF). A proper control allocation scheme is implemented, see [13], to allow for optimal allocation of the effort without violating thruster capabilities.

The proposed on-line reinforcement learning algorithm which is based on least square policy iteration (LSPI) [15], [16], aims at the determination of a near-optimal path in the presence of realistic environmental wind and sea current disturbances. Simulation results show that the generated path is tracked successfully by the marine platform, which

K. Tziortziotis, K. Vlachos, and K. Blekas are with the Department of Computer Science and Engineering, University of Ioannina, 45110 Ioannina, Greece. (Email: {ktziortz; kostaswl; kblekas}@cs.uoi.gr)

978-1-4673-8345-5/16/$31.00 ©2016 IEEE

Fig. 1. The triangular marine platform.

is described not only by the kinematic but also by the dynamic model. Moreover, the dynamics and limitations of the actuation system are modeled into the simulation environment. One of the main advantages of this method is that it is model free, meaning that in the design process we did not use any explicit knowledge about the system model. This makes the method robust to model uncertainties and noise, as it is demonstrated in our results. Another advantage is that it can be implemented as an online learning algorithm which brings us a step further towards fully autonomous marine platform.

II. The Marine Platform

The marine platform is designed to assist in the deployment of a deep-sea cubic kilometer neutrino telescope. It consists of a triangular structure mounted on three hollow double-cylinders, one at each corner of the structure, see Fig. 1. The plane of the triangle is parallel to the sea surface. The cylinders provide the necessary buoyancy, as part of them is immersed in the water. The platform actuation is realized using three fully submerged pump-jets, located at the bottom of each cylinder. Diesel engines drive the pumps, while electro-hydraulic motors rotate the jets providing vectored thrust. A comprehensive description can be found in [13].

A. Kinematics

The main body of the structure has the shape of an isosceles triangle with side length $L_{AB} = L_{AC}$, and base length L_{BC}. The center of mass (CM) of the platform is at point G, see Fig. 2. We focus on the platform planar motion; actuation and control along the heave axis, and about the roll and pitch axes, are beyond the scope of this work. Under these assumptions, the kinematics equations of the plane motion are described by:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}\cos\psi & -\sin\psi & 0\\ \sin\psi & \cos\psi & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}u\\ v\\ r\end{bmatrix} \Rightarrow {}^{I}\dot{\mathbf{x}} = {}^{I}\mathbf{R}_{B}\,{}^{B}\mathbf{v} \quad (1)$$

In (1), x and y represent the platform CM inertial coordinates and ψ describes the orientation of the body-fixed frame $\{\mathrm{B}\}$, whose origin is at the platform CM; u and v are the surge and sway velocities respectively, defined in the body-fixed frame $\{\mathrm{B}\}$, and r is the yaw (angular) velocity of the platform.

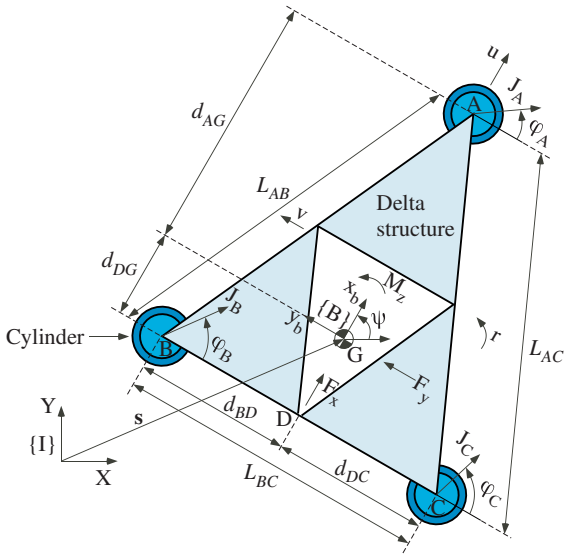


Fig. 2. A 2D representation of the triangular platform.

B. Dynamics

We consider three types of forces acting on the CM of the platform: (a) the control forces/ torque from the jets, (b) the hydrodynamic forces due to the motion of the cylinders with respect to the moving water, and (c) the disturbance forces/torque due to wind.

1) Control forces/torque: The jets can provide vectored thrust and thus more flexibility in control design. The J_A, J_B, and J_C in Fig. 2 denote the magnitudes of the thrusts while ϕ_A, ϕ_B, and ϕ_C denote the force directions. These thrusts provide control resultant forces in x_b and y_b axes, the F_x and F_y respectively acting on the CM, and a torque M_z about z_b, according to the linear transformation:

$${}^{B}\mathbf{n}_c = [F_x, F_y, M_z]^T = \mathbf{B}\,{}^{B}\mathbf{f}_c \quad (2)$$

$$\mathbf{B} = \begin{bmatrix}1 & 0 & 0\\ 0 & -1 & -d_{AG}\\ 1 & 0 & -d_{DC}\\ 0 & -1 & d_{DG}\\ 1 & 0 & d_{DC}\\ 0 & -1 & d_{DG}\end{bmatrix}^T, \quad {}^{B}\mathbf{f}_c = \begin{bmatrix}J_A\sin\phi_A\\ J_A\cos\phi_A\\ J_B\sin\phi_B\\ J_B\cos\phi_B\\ J_C\sin\phi_C\\ J_C\cos\phi_C\end{bmatrix} \quad (3)$$

where ${}^{B}\mathbf{n}_c$ is the control force/torque vector, and the dimensional parameters in $\mathbf{B}$ are defined in Fig. 2. The vector ${}^{B}\mathbf{f}_c$ can be retrieved by the pseudoinversion of $\mathbf{B}$ in 2. The desired jet thrust and direction are calculated according to,

$$J_i = \sqrt{(f_i\sin\phi_i)^2 + (f_i\cos\phi_i)^2} \quad (4)$$

$$\phi_i = \arctan(f_i\sin\phi_i, f_i\cos\phi_i) \quad (5)$$

where $i = A, B, C$. Note that the desired jet thrust and direction cannot be applied immediately due to actuator dynamics

and limitations. A detailed description of the dynamics and limitations of the actuation system of the platform can be found in [13].

2) Hydrodynamic forces: Assuming a constant or slowly varying water velocity, the hydrodynamic force acting on each cylinder includes two terms. The first term is the added mass force, which is a linear function of the acceleration of each cylinder. The second term is the drag force, which is a quadratic function of the relative velocity between the water and each cylinder, see [17]. As an example, the normal to the axis of each cylinder force on the double-cylinder structure at point A, expressed in body-fixed frame $\{\mathrm{B}\}$ is given by:

$$\begin{aligned} {}^{B}\mathbf{f}_{h,A} = C_a\pi\rho_w[R_{uc}^2(H_{uc}-h)+R_{lc}^2H_{lc}](-{}^{B}\mathbf{a}_A)+ \\ C_d\rho_w[R_{uc}(H_{uc}-h)+R_{lc}H_{lc}] \\ \|({}^{B}\mathbf{v}_{cur}-{}^{B}\,\mathbf{v}_A)\|({}^{B}\mathbf{v}_{cur}-{}^{B}\,\mathbf{v}_A) \end{aligned} \quad (6)$$

where ρ_w is the water density, C_a is the added mass coefficient, and C_d the drag coefficient. ${}^{B}\mathbf{v}_A$ and ${}^{B}\mathbf{a}_A$ are the velocity and acceleration of cylinder A respectively expressed in the body-fixed frame. ${}^{B}\mathbf{v}_{cur}$ denotes the sea current velocity expressed in the body-fixed frame. The parameters h, R_{uc}, H_{uc}, R_{lc}, and H_{lc} denote the height of the cylinder above the water surface, and the radius and height of the upper and lower cylinder sections respectively. The hydrodynamic forces on A given by Eq. 6 result in a force acting on the platform CM and a moment about it, i.e.,

$${}^{B}\mathbf{q}_{h,A} = [{}^{B}\mathbf{f}_{h,A}^T, ({}^{B}\mathbf{s}_{A/G}\times{}^{B}\,\mathbf{f}_{h,A})^T]^T \quad (7)$$

where ${}^{B}\mathbf{s}_{A/G}$ is the position of point A with respect to G expressed in $\{\mathrm{B}\}$, see Fig. 2. All terms that are a quadratic function of the velocity of the platform are collected in vector,

$${}^{B}\mathbf{q} = [f_x, f_y, n_z]^T \quad (8)$$

3) Environmental disturbances: We define the disturbance vector ${}^{B}\mathbf{q}_{wind}$, which represents wind generated disturbance forces and torque, see [18]. Integrating Gaussian white noise produces the inertial wind velocity magnitude and direction used in the simulations, see Fig. 3. The wind velocity magnitude, $v_w(t)$, is limited such that $v_w(t) \leq 7.9$ m/s (15 kn or 4 Beaufort).

The sea current induced forces and moments are included in the dynamic equations of motion by representing (6) in terms of sea current velocity ${}^{B}\mathbf{v}_{cur}$. The inertial sea current velocity magnitude and direction used in the simulations are produced by integrating Gaussian white noise, see Fig. 4. The sea current velocity magnitude, $v_c(t)$, is limited such that $v_c(t) \leq 0.5$ m/s.

4) Equations of motion of the platform: Using the above preliminaries, and assuming that the CM of the platform is at the triangles centroid, we derive the planar equations of motion of the platform, in $\{\mathrm{B}\}$:

$$\mathbf{M}{}^{B}\dot{\mathbf{v}} = {}^{B}\mathbf{q} + {}^{B}\mathbf{q}_{wind} + {}^{B}\mathbf{n}_c \quad (9)$$

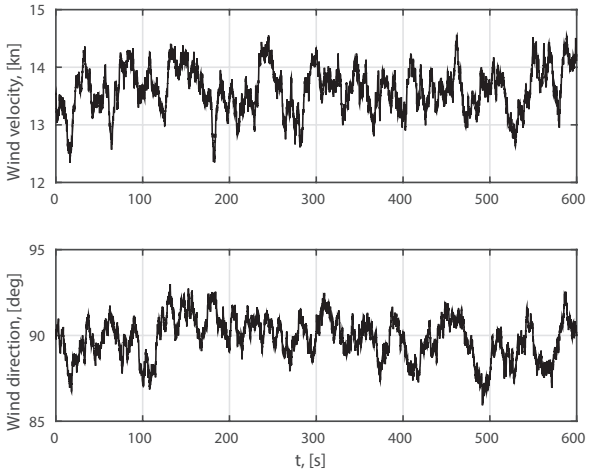


Fig. 3. Wind disturbances used in simulations.

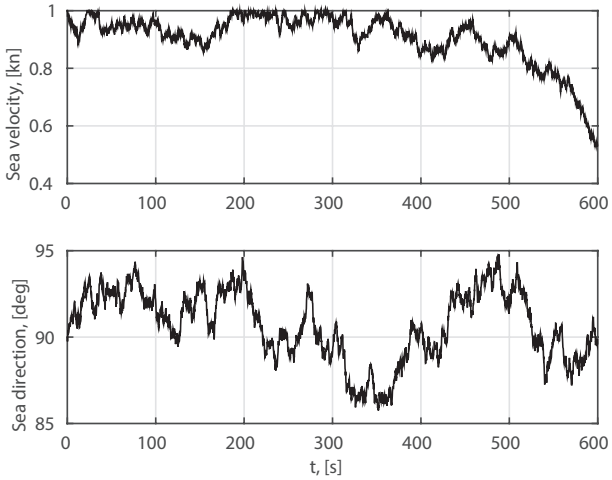


Fig. 4. Sea current velocity and direction.

$$\mathbf{M} = \begin{bmatrix} m-3m_a & 0 & 0 \\ 0 & m-3m_a & 0 \\ 0 & 0 & m_{33} \end{bmatrix} \quad (10)$$

$$m_{33} = I_{zz} - (d_{AG}^2 + 2d_{BD}^2 + 2d_{DG}^2)m_a \quad (11)$$

$$m_a = -C_a\pi\rho_w[R_{uc}^2(H_{uc}-h)+R_{lc}^2H_{lc}] \quad (12)$$

where m is the mass of the platform, m_a is its added mass, and I_{zz} is its mass moment of inertia about the z_b axis. The model described by (9) is used in all simulation runs presented in Section IV.

It must be noted that in the proposed RL-based framework, presented in the next Section, only the x, y inertial coordinates are considered, and consequently the resulted action is related only to the translation of the marine platform. However, the rotation of the platform is controlled independently according to a PD controller that aims at stabilizing the platform orientation to a desired angle in the presence of realistic wind disturbances. The desired orientation of the platform coincides with the direction of the wind. This configuration results to reduced disturbance forces/torque,

due to the reduction of the projected area to the wind. Consequently, the torque M_z is calculated according to,

$$M_z = K_p(\psi_w - \psi) - K_d r \tag{13}$$

where K_p and K_d are the controller gains, and ψ_w denotes the direction of the wind.

Nevertheless, including the orientation ψ of the body-fixed frame to the set of state features in our RL-based framework constitutes a subject for future research study.

III. Reinforcement Learning for Autonomous Marine Vehicle Navigation

The RL framework considers that the environment is modeled as a *Markov decision process* (MDP). An MDP can be described as a five-tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is a set of states; $\mathcal{A}$ a set of actions; $P : \mathcal{S} \times \mathcal{A} \times S \rightarrow [0, 1]$ is a Markovian transition model that specifies the probability $P(\boldsymbol{s}'|\boldsymbol{s}, a)$ of transition to state $\boldsymbol{s}'$ when taken an action a in state $\boldsymbol{s}$; $R : \mathcal{S} \rightarrow \mathbb{R}$ is the reward function for a state-action pair; and $\gamma \in (0, 1)$ is the discount factor for future rewards. A *stationary policy* $\pi : \mathcal{S} \rightarrow \mathcal{A}$ is a mapping from states to actions and denotes a mechanism for choosing actions. An *episode* is a sequence of transitions: $(\boldsymbol{s}_1, a_1, r_1, \boldsymbol{s}_2, \ldots)$. As noted before in our application we have considered that the state space consists of two platform inertial coordinates, i.e. $\boldsymbol{s} = (x, y)$, while the action is related to the translation of the marine platform.

The Q-function $Q : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ of the policy π gives for every state-action pair $(\boldsymbol{s}, a)$ the expected return when starting in s applying action a and following π thereafter. Q-values can be evaluated by solving the following set of linear Bellman equations:

$$Q^{\pi}(\boldsymbol{s}, a) = R(\boldsymbol{s}, a) + \gamma \sum_{\boldsymbol{s}' \in \mathcal{S}} P(\boldsymbol{s}'|\boldsymbol{s}, a) \max_{a} Q(\boldsymbol{s}', a). \tag{14}$$

The objective of RL problems is to estimate an optimal policy π^* by choosing actions that yield the optimal action-state value function Q^*: $\pi^*(\boldsymbol{s}) = \arg\max_a Q^*(\boldsymbol{s}, a)$.

A common choice is to consider linearly parameterized Q-function approximators. This can be done using a set of k basis functions $\phi(\boldsymbol{s}, a) = [\phi_1(\boldsymbol{s}, a), \ldots, \phi_k(\boldsymbol{s}, a)]^{\top}$:

$$Q(\boldsymbol{s}, a) = \boldsymbol{\phi}(\boldsymbol{s}, a)^{\top} \boldsymbol{w} = \sum_{j=1}^{k} \phi_j(\boldsymbol{s}, a) w_j \ , \tag{15}$$

where $\boldsymbol{w} = (w_1, \ldots, w_k)$ is a vector of weights which are unknown and must be estimated so as to minimize the approximation error. The selection of the basis functions is very important and must be chosen to encode properties of the state and action relevant to the proper determination of the Q values.

In our work we have used RBF basis functions:

$$\phi_j(\boldsymbol{s}) = \exp(-\beta_j \|\boldsymbol{s} - c_j\|^2) \ , \tag{16}$$

for a given collection of centers c_j and precision (inverse variance) β_j. Note that we have considered common precision to all k functions, i.e. $\beta_j = \beta$. In order to obtain the parameters of these k basis functions we have following the idea of *tile coding* that is based on (uniformly) partitioning the state space on k non-overlapping regions and obtaining their geometrical features (center and width). An alternative way is to employ Fourier basis functions [19] for value function approximation, described as (nth order Fourier basis): $\phi_j(\boldsymbol{s}) = \cos(\pi c_j^{\top} \boldsymbol{s})$. The latter provides a non-parametrized kernel formulation. However, in both cases the number and the structure of k basis functions remain fixed during the learning process.

Policy iteration is a dynamic programming algorithm, which starts with an arbitrary policy and steadily improves it. It discovers the optimal policy by generating a sequence of monotonically improving policies. The policy iteration algorithm manipulates the policy directly instead of finding it via the value function, as happens in the case of value iteration. Policy iteration consists of two successive, interactive phases: the *policy evaluation* where the value function of policy π is computed and the *policy improvement*. The above two phases are executed iteratively until policy π cannot be further improved. In this case, the policy iteration algorithm converges to the optimal policy π^*.

Least-Squares Policy Iteration (LSPI) [15] is a batch approximate policy iteration algorithm. It adopts the approximate policy-iteration framework and uses a *model-free* version of the least-squares temporal difference learning (LSTD). The action-value function Q, is approximated instead of the state-value function, while action selection and policy improvement are permitted without the need of any prior knowledge of the environment dynamics. In its original form, the LSPI is an off-line algorithm and requires a set of training examples: $D = \{s_i, a_i, r_i, s_i' | i = 1, \ldots, n\}$, which are used at each iteration in order to evaluate the derived policies. During the policy evaluation step, the matrix A (size $k \times k$) and the vector b (size $k \times 1$), are computed following the previously learned policy π, respectively, as follows:

$$A = \sum_{i=1}^{n} \phi(s_i, a_i)(\phi(s_i, a_i) - \gamma \phi(s_i', \pi(s_i')))^{\top}, \tag{17}$$

$$b = \sum_{i=1}^{n} \phi(s_i, a_i) r_i. \tag{18}$$

At the policy improvement step, matrix A and vector b are used in order to yield an improved policy. In this way, the least-squares projection error for the state-value function Q, is minimized as:

$$w = A^{-1} b. \tag{19}$$

The whole procedure is implemented iteratively, until a convergence criterion is satisfied. The model parameters are initialized arbitrarily or are set to 0.

A number of approaches have been proposed for rendering the LSPI framework in an online mode. Recently, an online variant of the LSPI has been presented in [16], where the ϵ-greedy exploration scheme is adopted. This approach is based on an online scheme, where at each time step t an action is selected greedily, based on the estimated action-value

function with probability $1-\epsilon_t$ ($\epsilon_t \in [0,1]$), while a uniform random exploratory action is applied with probability ϵ_t. Initially, the parameter ϵ_0 is set to a large value (e.g., $\epsilon_0 = 1$), while it decays exponentially over time with a decay rate $\epsilon_d \in (0,1)$. In the particular scheme, policy improvement can be implemented after a number of consecutive transitions. In our study we have followed this scheme where policy improvement were performed either every 30 transitions, or at the end of each episode. Another interesting online approach is the one presented in [20]. In that work, the R-max exploration scheme is integrated in the LSPI learning algorithm.

IV. Simulation Results

We have studied the performance of the proposed method using several simulated experiments. The simulation environment has been implemented using the MATLAB software package. The environment includes the kinematic and dynamic model of the marine platform, and simulated wind and sea current disturbances, Moreover, the dynamics and limits of the actuators are also implemented into the simulation environment. In all cases the integration time step was set to $dt = 0.2$.

During the simulation runs we have designed grid maps as test environments that contain obstacles of various difficulty. The objective of the marine vehicle in this task is to find a steady landmark with a minimum number of steps starting from a particular position and performing a finite number of actions. The map was completely unknown to the platform and the study was focused on the proposed method's ability to generate a physically realizable path at a reasonable computational cost under its motion constraints and the external disturbances.

Figures 5 and 6 illustrates two experimental maps of different complexity, where the landmark is presented as a small rectangle. Also, the (stochastic) direction of the wind was chosen to be 90 degrees (std 1) and the (stochastic) wind velocity magnitude 7 kn (std 0.04). During the learning process a new episode starts when one of the following incidents comes first: the maximum allowed number of steps per episode is expired (in our case was set to 1000), an obstacle is hit, or the target is reached. At each time step, the marine vehicle receives an immediate reward of -1, except in the case that an obstacle is hit where the received reward is -100. Finally, when the target is found a reward of 1 is returned.

For the construction of the basis functions we have used an equidistant 5×5 grid of RBFs over the $2D$ state space, after performing a $[0,1]$ normalization. Therefore, a number of $25 \times K$ RBFs were constructed, where K denotes the number of different actions. The action space corresponds to the control forces F_x and F_y. Even if in this study we have used a constant force magnitude of 15000 N, we could alternatively consider a discrete set of different level magnitude values. This may offer the flexibility to make the proposed intelligent agent more adaptive and robust to environmental situations and external disturbances. On

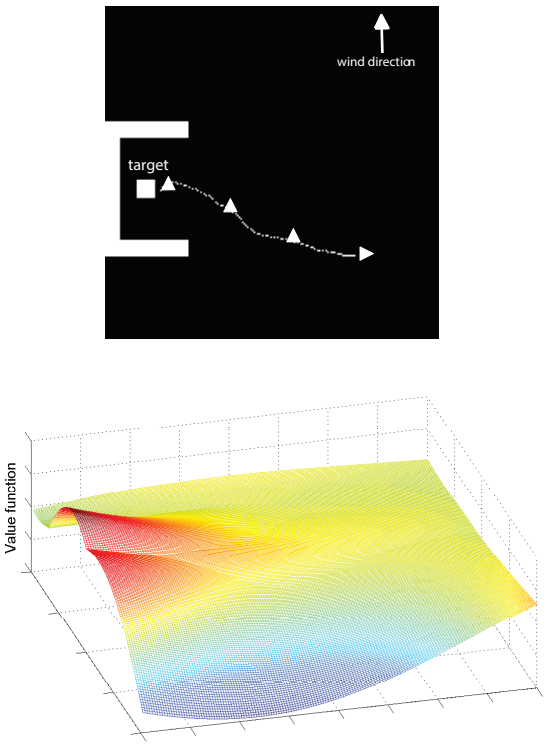


Fig. 5. Optimum policies and navigation path in the case of two test environments.

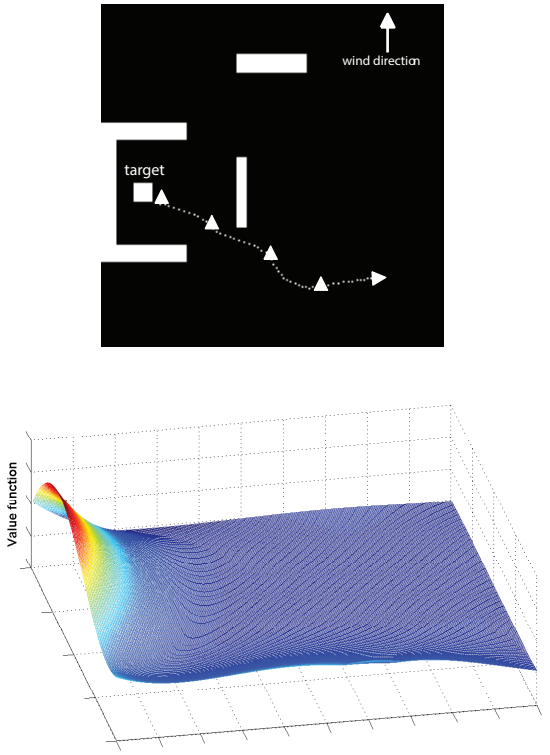
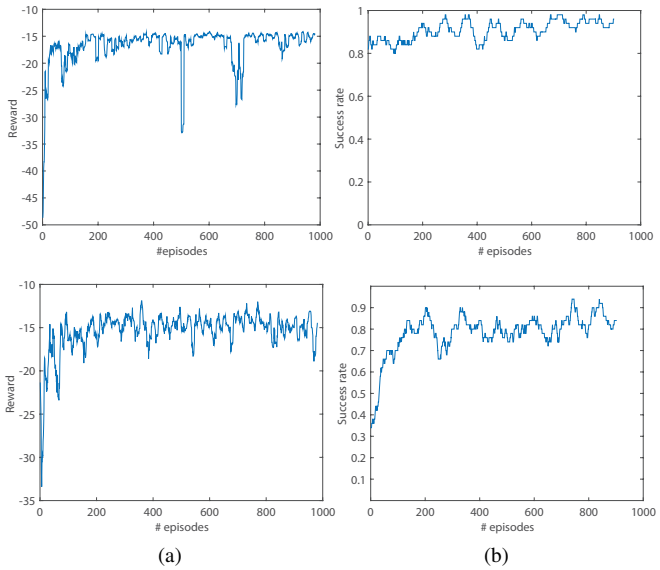


Fig. 6. Optimum policies and navigation path in the case of two test environments.



(a) (b)

Fig. 7. Learning curves as depicted from the RL model in the case of two test environments. The estimated reward value (a) and the frequency of target found (b) per episode are shown.

the other hand we have considered 4 values for the force direction, leading to a final discrete action space that consists of $K = 4$ possible actions). Finally, in all domains, the discount factor γ was set equal to 0.99,

The depicted results are illustrated in Figures 5, 6 that give the final solution paths from the selected steady state to the goal state (target), consisting of a number of around 100 steps. We also give in Figure 7 the depicted learning curves of the proposed RL model. In particular the progress of the calculated mean reward value of the last 50 episodes is shown (Fig. 7 (a)), as well as the mean success rate of the method, i.e. the frequency of finding the target (computed from the last 50 episodes). Note that the RL converges to the optimal solution after (approximately) 200 episodes in either case. Finally, we also present in Figures 5, 6 the learned policies of the method in both grids after 1000 episodes, in terms of the calculated values of the Q-function. As it can be observed, the proposed method successfully found sub-optimum policies in both test environments.

It must be noted that, according to our simulation study, the number and the structure of the basis functions constitutes a significant issue to the performance of out method. Even if the number of 25 basis functions gave a satisfactory behavior in both environments, their proper determination and adaptation suggests a future direction in our study.

V. Conclusions and Discussion

In this study we presented an autonomous navigation framework of a triangular floating marine platform involving a reinforcement learning scheme. The method tries to learn the policy function and estimate a target position according to a least-squares mechanism that uses RBF kernel functions. The system is over-actuated and allows on-line learning capabilities. Simulated results illustrated its performance under environmental disturbances.

We plan to verify the proposed algorithm in the future in a more complex simulated environment including additional disturbance sources, i.e. environmental disturbance forces/torque due to waves, and state measurement noise due to GPS measurement error. Finally, a significant issue in constructing reinforcement learning agents in Markov decision processes is how to design efficient feature spaces. Our primitive aim is to address this challenge in the marine vehicle navigation problem under two aspects: a) to study alternative schemes for constructing appropriate basis functions, and b) to model the marine navigation task with partially observable Markov decision processes (POMDPs).

References

[1] M. Seto, *Marine Robot Autonomy*. Spinger-Verlag, 2013.

[2] G. Antonelli, *Underwater Robots*, ser. Springer Tracts in Advanced Robotics. Springer, 2014, vol. 96.

[3] R. Sutton, "Generalization in reinforcement learning: Successful examples using sparse coarse coding," in *Advances in Neural Information Processing Systems 8*. MIT Press, 1996, pp. 1038–1044.

[4] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. MIT Press Cambridge, USA, 1998.

[5] J. N. Tsitsiklis and R. Sutton, "Asynchronous stochastic approximation and q-learning," pp. 185–202, 1994.

[6] Y. Engel, S. Mannor, and R. Meir, "Reinforcement learning with gaussian process," in *International Conference on Machine Learning*, 2005, pp. 201–208.

[7] N. Tziortziotis and K. Blekas, "Model-based reinforcement learning using online clustering," in *IEEE Inern. Conference on Tools with Artificial Intelligence (ICTAI)*, 2012, pp. 712–718.

[8] N. Tziortziotis, C. Dimitrakakis, and K. Blekas, "Linear bayesian reinforcement learning," in *Inern. Joint Conference on Artificial Intelligence (IJCAI)*, 2013, pp. 1721–1728.

[9] ——, "Cover tree bayesian reinforcement learning," *Journal of Machine Learning Research*, vol. 15, pp. 2313–2335, 2014.

[10] M. Carreras, J. Yuh, J. Battle, and P. Ribao, "A behavior-based scheme using reinforcement learning for autonomous underwater vehicles," *IEEE Journal of Ocean Engineering*, vol. 30, pp. 416–427, 2005.

[11] H. Kawano, "Method for applying reinforcement learning to motion planning and control of under-actuated underwater vehicle in unknown non-uniform sea flow," in *IEEE International conference on Intelligent Robots and Systems (IROS)*, 2005, pp. 996–1002.

[12] B. Yoo and J. Kim, "Path optimization for marine vehicles in ocean ccurrents using reinforcement learning," *Journal of Marine Science and Technology*, pp. 1–10, 2015.

[13] K. Vlachos and E. Papadopoulos, "Modeling and control of a novel over-actuated marine floating platform," *Ocean Engineering*, vol. 98, pp. 10–22, 2015.

[14] A. Tsopelakos, K. Vlachos, and E. Papadopoulos, "Backstepping control with energy reduction for an over-actuated marine platform," in *IEEE Intern. Conference on Robotics and Automation (ICRA)*, 2015, pp. 553–558.

[15] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *Journal of Machine Learning Research*, vol. 4, pp. 1107–1149, 2003.

[16] L. Busoniu, D. Ernst, B. D. Schutter, and R. Babuska, "Online least-squares policy iteration for reinforcement learning control," in *American Control Conference (ACC)*, 2010, pp. 486–491.

[17] S. Hoerner, *Fluid-Dynamic Drag*. Hoerner Publications, 1965.

[18] T. Fossen, *Guidance and Control of Ocean Vehicles*. John Wiley and Sons, 1994.

[19] G. Konidaris, S. Osentoski, and P. Thomas, "Value function approximation in reinforcement learning using the fourier basis," in *AAAI Conf. on Artificial Intelligence*, 2011, pp. 380–385.

[20] L. Li, M. Littman, and C. Mansley, "Online exploration in least-squares policy iteration," in *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2009, pp. 733–739.