International Journal on Artificial Intelligence Tools
Vol. 27, No. 4 (2018) 1860005 (20 pages)

DOI: 10.1142/S0218213018600059

World Scientific
www.worldscientific.com

Motion Planning with Energy Reduction for a Floating Robotic Platform Under Disturbances and Measurement Noise Using Reinforcement Learning

Konstantinos Tziortziotis
Department of Computer Science & Engineering
University of Ioannina, Ioannina, Greece
ktziortz@cs.uoi.gr

Nikolaos Tziortziotis
Computer Science Laboratory (LIX), Ecole Polytechnique, Paris
ntziortziotis@lix.polytechnique.fr

Kostas Vlachos* and Konstantinos Blekas†
Department of Computer Science & Engineering
University of Ioannina, Ioannina, Greece
**kostaswl@cs.uoi.gr*
†*kblekas@cs.uoi.gr*

Received 4 September 2017
Accepted 23 January 2018
Published 27 June 2018

This paper investigates the use of reinforcement learning for the navigation of an over-actuated, i.e. more control inputs than degrees of freedom, marine platform in unknown environment. The proposed approach uses an online least-squared policy iteration scheme for value function approximation in order to estimate optimal policy, in conjunction with a low-level control system that controls the magnitude of the linear velocity, and the orientation of the platform. Primary goal of the proposed scheme is the reduction of the consumed energy. To that end, we propose a variable reward function that depends on the energy consumption of the platform. We evaluate our approach in a complex and realistic simulation environment and report results concerning its performance on estimating optimal navigation policies under different environmental disturbances, and position GPS measurement noise. The proposed framework is compared, in terms of energy consumption, to a baseline approach based on virtual potential fields. The results show that the marine platform successfully discovers the target point following a sub-optimal path, maintaining reduced energy consumption.

Keywords: Reinforcement learning; autonomous navigation; over-actuated marine vehicle; energy reduction.

†Corresponding author

1. 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 consumption, etc. Robust motion planning algorithms for mobile robots consider stochasticity in the dynamic model of the vehicle and the environment.

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 modelled 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 return 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 suitable 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, a variety of value function approximation schemes have also been presented, including: Gaussian processes,[6] online clustering techniques which learn basis function sets from experience,[7] and Bayesian reinforcement learning approaches for the exact inference of unknown dynamics in continuous state spaces,[8–10] etc.

In the literature there are some marine robots applications, mostly involving autonomous underwater vehicles (AUV), using reinforcement learning, see for a survey in Refs. 1 and 11. In Ref. 12 for example a neural networks-based reinforcement learning scheme is presented for high-level control of AUV's. In Ref. 13 another approach is proposed for motion planning of under-actuated AUV in unknown non-uniform sea flow. A recent work for marine vehicle navigation in Ref. 11 uses a path planning algorithm for under-actuated marine vehicles under ocean current disturbances based on reinforcement learning. In this case, the marine vehicle is described only by the kinematic model.

Fig. 1. The triangular marine platform.[22]

Our work focuses on the development of an intelligent navigation mechanism based on reinforcement learning, for the over-actuated autonomous triangular marine platform "Vereniki" shown in Fig. 1. In addition, the magnitude of the linear velocity, and the orientation of the platform is controlled by a low-level control scheme. The detailed model of the platform can be found in Ref. 14.

Various control schemes for the autonomous dynamic positioning of the platform have been proposed in previous papers. A Model-based controller has been presented in Ref. 14. A linear MPC, and a Backstepping controller have also been presented in Refs. 15 and 16 respectively. Nevertheless, in all these approaches, the target configuration of the platform was known, and the goal of the control was not the motion planning in the presence of unknown obstacles, but the dynamic positioning of the platform on that configuration.

Here, we examine the problem of the determination of a desired path in an unknown environment. 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. Thus, a non-trivial problem arises concerning the optimal use of the control inputs. To solve this problem, a proper control allocation scheme is implemented to allow for optimal allocation of the effort without violating thruster capabilities, see Ref. 14.

The proposed on-line reinforcement learning algorithm which is based on least square policy iteration (LSPI),[17,18] aims at the determination of a near-optimal path in the presence of realistic environmental wind, and wave disturbances, and position measurement noise. Simulation results show that the generated path is tracked successfully by the marine platform, which is described not only by the kinematic but also by the dynamic model. 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.

The novel material in this paper, compared to our previous work in Refs. 19 and 20 where initial results were reported, can be summarized as follows:

- An important feature of the proposed scheme is the determination of a sub-optimal policy that leads to the reduction of the energy consumption. To that end, we propose a reward function that depends on the energy consumption of the platform at each step.
- The action space of the RL agent now includes the direction of the velocity of the marine platform, instead of the thrust acting on the platform used in our previous work. This simplifies the discovery process of the RL agent optimal policy.
- A low-level controller ensures that the magnitude of the linear velocity equals to an arbitrary predefined value, and the orientation of the platform equals to the direction of the wind, resulting in reduced disturbances.
- New extensive simulation results are presented in a more comprehensive, realistic, and challenging simulated environment.
- The proposed framework is compared, in terms of energy consumption, to a baseline approach based on virtual potential fields with a *complete known* map.
- The simulated environment includes a new workspace based on a real map taken from Google Maps, additional wind generated wave disturbance forces/torque, and position GPS measurement noise.

The remainder of this paper is organized as follows. Section 2 gives an overview of the marine platform and its control design issues, while Section 3 gives the reinforcement learning scheme. Simulation results are presented in Section 4 and we conclude with a discussion of future directions in Section 5.

2. The "Vereniki" Marine Platform

The marine platform "Vereniki" is designed to assist in the deployment of the deep-sea cubic kilometer neutrino telescope "NESTOR".[21] 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. Next, for the sake of completeness, a brief description of the kinematics and dynamics of the platform is presented. A more comprehensive description of the platform can be found in Ref. 14.

2.1. *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

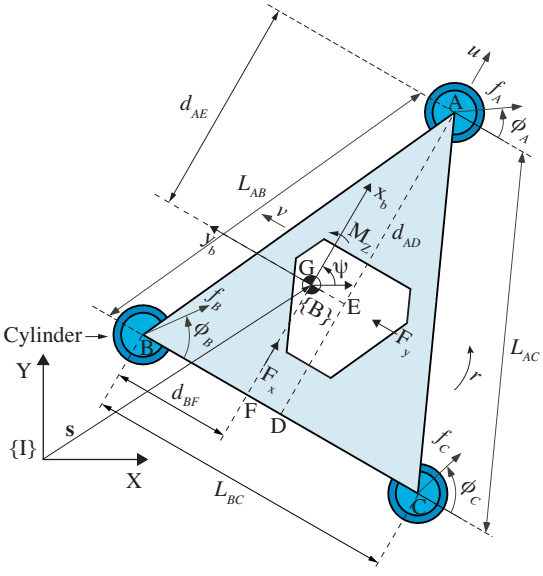


Fig. 2. A 2D representation of the triangular platform.[14]

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:

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

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

2.2. *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 water, and (c) the environmental disturbance forces/torque due to wind, and wind generated waves.

2.2.1. *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:

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

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

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,

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

$$\phi_i = \arctan(f_i \sin\phi_i, f_i \cos\phi_i) \tag{5}$$

where $i = A, B, C$.

2.2.2. *Hydrodynamic forces*

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 velocity of each cylinder, see Ref. 23. 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 {B} is given by:

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

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. 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 (6) result in a force acting on the platform CM and a moment about it, i.e.,

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

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

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

2.2.3. *Environmental disturbances and measurement noise*

We define the disturbance vector ${}^{B}\mathbf{q}_{env\_dist}$, which represents wind, and wind generated wave disturbance forces and torque. The wind induced forces (surge and sway) and moment (yaw), are calculated as,

$$ f_{x,wind} = 0.5 C_X(\gamma_R)\rho V_R^2 A_T \tag{9}$$

$$ f_{y,wind} = 0.5 C_Y(\gamma_R)\rho V_R^2 A_L \tag{10}$$

$$ n_{z,wind} = 0.5 C_T(\gamma_R)\rho V_R^2 A_L L \tag{11}$$

where C_X and C_Y are force coefficients and C_T is a moment coefficient. These coefficients are functions of the relative angle, γ_R, between the wind and platform direction, and are taken from tables. ρ is the density of air, A_T and A_L are the transverse and lateral projected area, and L is the overall length of the platform. V_R is the relative wind speed, given in knots. Integrating Gaussian white noise produces the inertial wind velocity magnitude and direction used in the simulations.

The wind velocity magnitude, $v_w(t)$, is limited such that $v_w(t) \leq 7.9$ m/s (15 kn or 4 Beaufort). In our simulated environment, the wave signals are functions of the simulated wind signals. A detailed description of the generation of wind, and wind depended wave signals, and the calculation of the respected disturbance forces and torque can be found in Ref. 24. Example time series of the simulated wind signal are shown in Figs. 3(a) and 3(b). In both figures, the initial wind velocity magnitude is equal to 4 kn. The initial wind velocity direction is equal to 0° and 180° respectively.

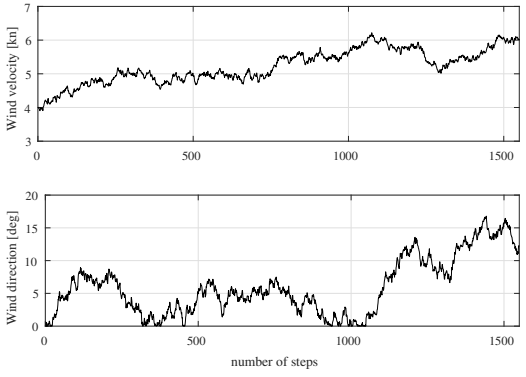
Using the above preliminaries, and assuming that the CM of the platform is at the triangle's centroid, we derive the planar equations of motion of the platform, in {B}:

$$ \mathbf{M}{}^{B}\dot{\mathbf{v}} = {}^{B}\mathbf{q} + {}^{B}\mathbf{q}_{env\_dist} + {}^{B}\mathbf{n}_c \tag{12}$$

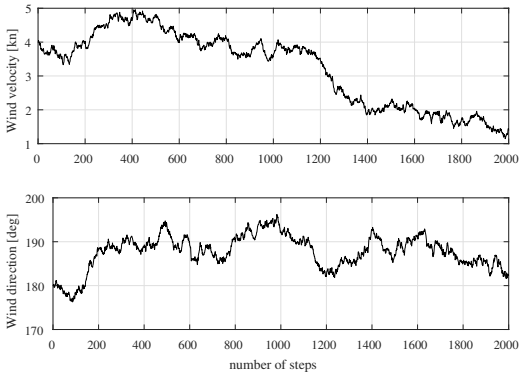
$$ \mathbf{M} = \begin{bmatrix} m - 3m_a & 0 & 0 \\ 0 & m - 3m_a & 0 \\ 0 & 0 & m_{33} \end{bmatrix} \tag{13}$$

$$ m_{33} = I_{zz} - (d_{AG}^2 + 2d_{BD}^2 + 2d_{DG}^2)m_a \tag{14}$$

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.[14]



(a)



(b)

Fig. 3. (a) Wind signal with initial wind velocity magnitude equal to 4 kn, and initial wind velocity direction equal to 0°. (b) Wind signal with initial wind velocity magnitude equal to 4 kn, and initial wind velocity direction equal to 180°.

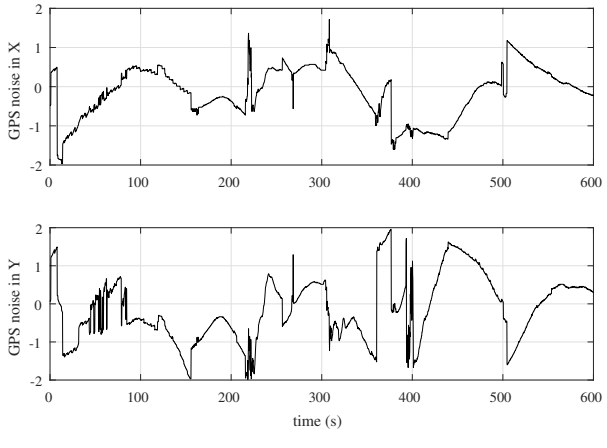


Fig. 4. Sample time series of the position GPS measurement noise.

Usually, the position and orientation of the platform are provided by GPS receivers that introduce measurement noise to the input. In our simulated environment, the receivers have an accuracy of ±2 m with an update frequency of 5 Hz. In order to simulate the measurement noise, we used noise from real GPS receivers. This is achieved by subtracting from the GPS readings the known position of the antennas. Then, the measurement noise is superimposed to the simulated trajectory of the platform. A sample time series of the extracted GPS noise used in the simulation runs is shown in Fig. 4.

2.2.4. *Velocity and rotation control of the platform*

As shown in (2), the control vector ${}^{B}\mathbf{n}_c$ includes the forces F_x and F_y contributing to the translation of the marine platform in x_b and y_b axes, and the torque M_z contributing to the rotation of the platform about z_b. This vector is the output of a controller scheme with two independent closed loops. The first closed loop realize a velocity controller where the input is the desired velocity of the floating platform (as commanded by the RL-agent), and the output is the desired forces according to

$$[F_x, F_y]^\top = K_{p1}(\begin{bmatrix}\dot{x}_{des} & \dot{y}_{des}\end{bmatrix}' - \begin{bmatrix}\dot{x} & \dot{y}\end{bmatrix}') \tag{15}$$

where K_{p1} is the controller gain related to the desired forces calculation, and $\dot{x}_{des}$ and $\dot{y}_{des}$ are the desired inertial linear velocities. The second closed loop is a PD controller where the input is the desired orientation of the floating platform, and the output is the desired torque according to

$$M_z = K_{p2}(\psi_w - \psi) - K_d r \tag{16}$$

where K_{p2}, and K_d are the controller gains related to the desired torque calculation, and ψ_w denotes the direction of the wind. As (16) suggests, 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 of the platform to the wind. The desired inertial linear velocities, $\dot{x}_{des}$ and $\dot{y}_{des}$, are calculated using a constant velocity magnitude defined by design, and a desired direction, which is the action of the RL agent, (see next section).

3. Reinforcement Learning for Autonomous Marine Vehicle Navigation

According to the Reinforcement Learning (RL) framework 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: $\mathcal{S} \times \mathcal{A} \times S \to [0, 1]$ 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: $\mathcal{S} \to \mathbb{R}$ is the reward function for a state-action pair; and $\gamma \in (0, 1)$ is the discount factor for future rewards. A *stationary policy* $\pi : \mathcal{S} \to \mathcal{A}$ is a mapping from states to actions and denotes

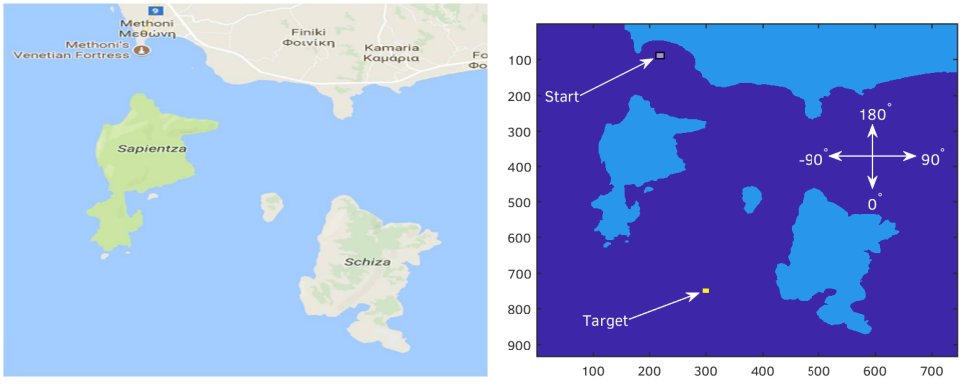


Fig. 5. A map of the site used as the test environment in our experiments.

a mechanism for choosing actions. An *episode* is given as a sequence of transitions: $\{(\boldsymbol{s}_1, a_1, r_1), (\boldsymbol{s}_2, a_2, r_2), \ldots\}$.

In our application we have considered the two platform inertial coordinates, $\boldsymbol{s} = (s_x, s_y)$, as the state variables, while the action consists of five (5) discrete values: $\mathcal{A} = \{-90°, -45°, 0°, 45°, 90°\}$ that correspond to five different directions of the marine platform velocity (see Fig. 5). Including the orientation ψ of the marine platform to the set of state features in our RL-based framework constitutes a subject for future research study.

The Q-function Q: $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$ of the policy π gives for every state-action pair $(\boldsymbol{s}, a)$ the expected discounted return received by starting from state s and taking action a following the policy π:

$$Q^{\pi}(\boldsymbol{s}, a) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(\boldsymbol{s}_t | \boldsymbol{s}_0 = \boldsymbol{s}.a_0 = a\right] \tag{17}$$

Estimating the optimal policy π^* is equivalent on choosing actions that yield the optimal action-state value function Q^*: $\pi^*(\boldsymbol{s}) = \arg\max_a Q^*(\boldsymbol{s}, a)$. Q-values can be evaluated by solving the following set of Bellman equations:

$$Q^{\pi}(\boldsymbol{s}, a) = R(\boldsymbol{s}, a) + \gamma \sum_{\boldsymbol{s}' \in \mathcal{S}} P(\boldsymbol{s}' | \boldsymbol{s}, a) \max_a Q(\boldsymbol{s}', a)\,. \tag{18}$$

In common RL domains with large or infinite state spaces the value function cannot be calculated explicitly. and so there is a need for function approximation. A common choice is to consider the linear model using a set of k basis functions $\phi(\boldsymbol{s}, a) = [\phi_1(\boldsymbol{s}, a), \ldots, \phi_k(\boldsymbol{s}, a)]^{\top}$:

$$Q(\boldsymbol{s}, a) = \boldsymbol{w}^{\top} \boldsymbol{\phi}(\boldsymbol{s}, a) = \sum_{j=1}^{k} w_j \phi_j(\boldsymbol{s}, a)\,, \tag{19}$$

where $\boldsymbol{w} = (w_1, \ldots, w_k)$ is a vector of linear coefficients 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. In our work we have used RBF basis functions:

$$\phi_j(\boldsymbol{s}) = \exp(-\beta_j \|\boldsymbol{s} - c_j\|^2)\,, \tag{20}$$

for a given collection of centers c_j and precision (*inverse variance*) β_j. The determination of these k basis functions was made by following the idea of *tile coding*, where the state space is (uniformly) partitioning into k non-overlapping regions and obtaining their geometrical centers, c_j. Note that we have considered common precision to all k functions, i.e. $\beta_j = \beta$.

Policy iteration is a dynamic programming algorithm, which starts with an arbitrary policy and steadily improves it. The policy iteration algorithm manipulates the policy directly instead of finding it via the value function, as happens in the case of value iteration. It 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)[17] is a batch approximate policy iteration algorithm that uses a *model-free* version of the least-squares temporal difference learning (LSTD). In its original form, the LSPI is an off-line algorithm and requires a set of training examples: $D = \{\boldsymbol{s}_i, a_i, r_i, \boldsymbol{s}'_i | i = 1, \ldots, n\}$, which are used at each iteration in order to evaluate the derived policies. During the policy evaluation step, the following matrix A (size $k \times k$) and the vector $\boldsymbol{b}$ (size $k \times 1$), are computed following the previously learned policy π:

$$A = \sum_{i=1}^{n} \phi(s_i, a_i)(\phi(s_i, a_i) - \gamma\phi(s'_i, \pi(s'_i)))^\top\,, \tag{21}$$

$$\boldsymbol{b} = \sum_{i=1}^{n} \phi(s_i, a_i) r_i\,. \tag{22}$$

At the policy improvement step, matrix A and vector $\boldsymbol{b}$ are used in order to yield an improved policy:

$$\boldsymbol{w} = A^{-1}\boldsymbol{b}\,. \tag{23}$$

The whole procedure is implemented iteratively, until a convergence criterion is satisfied.

An online variant of the LSPI has been presented in Ref. 18 that is based on the incremental nature of (21), (22). In particular, after making a movement in state space and receiving the tuple $(\boldsymbol{s}, a, r, \boldsymbol{s}')$, both quantities are updated as follows:

$$A^{new} = A^{old} + \phi(\boldsymbol{s}, a)(\phi(\boldsymbol{s}, a) - \gamma\phi(\boldsymbol{s}', \pi(\boldsymbol{s}')))^\top\,, \tag{24}$$

$$\boldsymbol{b}^{new} = \boldsymbol{b}^{old} + r\phi(\boldsymbol{s}, a)\,. \tag{25}$$

Special care was also made to manage the trade-off between exploration and exploitation since it has a significant impact to the quality of learned policy.[3,25] A common choice is to employ the ϵ-greedy exploration scheme,[26,3] where at each time step t an action is selected greedily, based on the estimated action-value function with probability $1-\epsilon_t$, while a random action is used with probability ϵ_t. Initially, the parameter ϵ_0 is set to a large value (e.g., $\epsilon_0 = 0.7$), while it decays exponentially over time with a decay rate of 0.99. Finally, in our particular scheme policy improvement was implemented after a number of 10 consecutive transitions, or at the end of each episode.

3.1. *The proposed reward function*

The goal of the proposed methodology is to achieve a sub-optimal policy of the marine platform with minimum energy consumption. Therefore, the proposed reward depends on the energy consumption at each step.

In particular, a *variable* reward is received given by $-\bar{E}/\bar{E}_{max}$, where $\bar{E}$ is the average energy consumption of the three jets during the step according to

$$\bar{E} = \frac{1}{3} \sum_{i=A,B,C} \int |p_i| \, dt \,. \tag{26}$$

$\bar{E}_{max}$ denotes the upper limit of $\bar{E}$, while p_i describes the input power for the ith jet during the step. In order to calculate the input power, we have used indicative numerical tables with the input power of real pump jets as a function of the produced thrust. The appropriate fitting of the numerical data resulted to the following polynomial that describes the input power of each jet as a function of the thrust:

$$p_i = 3.2930 f_i^4 + 2.1234 f_i^3 + 7.8610 f_i^2 + 0.0030 f_i + 0.1743 \tag{27}$$

where f_i is the thrust of the ith jet. Furthermore, in case of a collision with an obstacle or when the platform is outside the workspace, the received reward is $-\lambda\bar{E}_{max}$. Finally, when the target is found a positive reward equal to $\lambda\bar{E}_{max}$ is returned. During in our simulation runs we have used $\lambda = 10$. According to the previous, the mathematical expression of the reward function is the following:

$$R(\boldsymbol{s}, a) = \begin{cases} -\lambda\bar{E}_{max}, & \text{if hit obstacle or out of workspace;} \\ +\lambda\bar{E}_{max}, & \text{if target found;} \\ -\bar{E}/\bar{E}_{max}, & \text{otherwise.} \end{cases} \tag{28}$$

Since the energy consumption depends on the number of steps, the use of the reward function (28) leads to simultaneously obtain a sub-optimal path in terms of the number of steps. Note that in our previous work the reward function was not variable and the RL optimization was implemented only in terms of the number of steps.[19,20]

4. Experimental 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, the environmental wind, and wave disturbances, and the GPS measurement noise. In all cases the integration time step was set to $dt = 0.2$, and the RL agent step is equal to 50 integration time steps. For validation purposes, we compare the proposed methodology with a baseline approach.

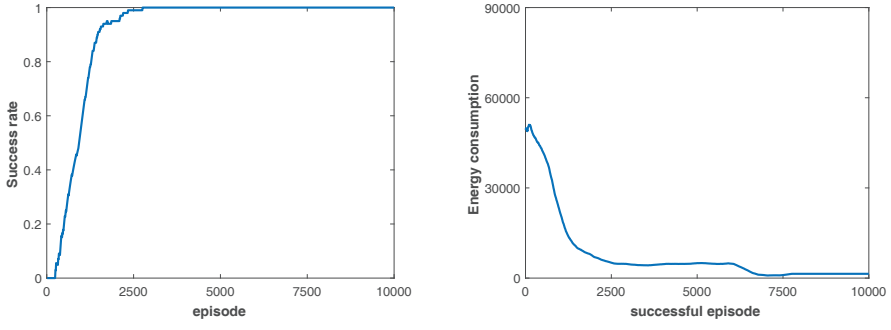
As mentioned in Section 2, the "Vereniki" platform is initially designed to assist in the deployment of the deep-sea cubic kilometer neutrino telescope "NESTOR". The "NESTOR" Project (Neutrino Extended Submarine Telescope with Oceanographic Research Project) was an international scientific collaboration whose target was the deployment of a neutrino telescope on the sea floor of Pylos and Methoni, South Peloponnese, Greece.[21,22] Therefore, in the simulation runs we have used a map of the area south of Methoni, as taken by the Google maps. The objective of the marine vehicle is to find a steady landmark (rectangular target), avoiding the physical obstacles in the area (islets), with minimum energy consumption, starting from the entrance of the port of Methoni. Figure 5 shows the original Google maps image and the corresponding binary image of size 744×933 pixels that represents our test environment. Note that 1 pixel corresponds to 15 meters.

In any case, 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.

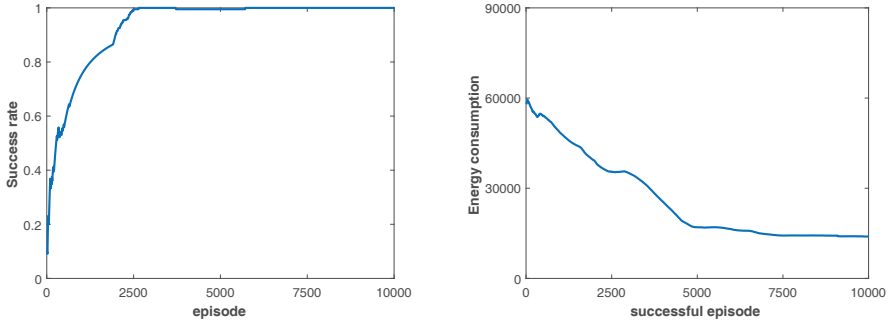
Several experiments were made in this simulated environment considering a variety of wind conditions. In particular we have used three values of wind velocity magnitude ($1, 4, 7$ kn) and three wind directions ($0°$, $90°$, $180°$). During the learning process a new episode starts when one of the following incidents comes first: the maximum allowed number of steps per episode is expired (in our case was set to 1000), an obstacle is hit, the platform is outside the workspace, or the target is reached.

For the construction of the basis functions we have partitioned the 2-dimensional state space on M equal and non-overlapping regions where we estimate their center (c_j) for constructing RBF mean parameters. A typical value of the number of basis function used in our experimental study was $M = 100$ with a common precision (inverse variance) $\beta = 10$. The action space corresponds to the direction of the platform velocity (see Section 3). Finally, in all cases the discount factor γ was set equal to 0.99.

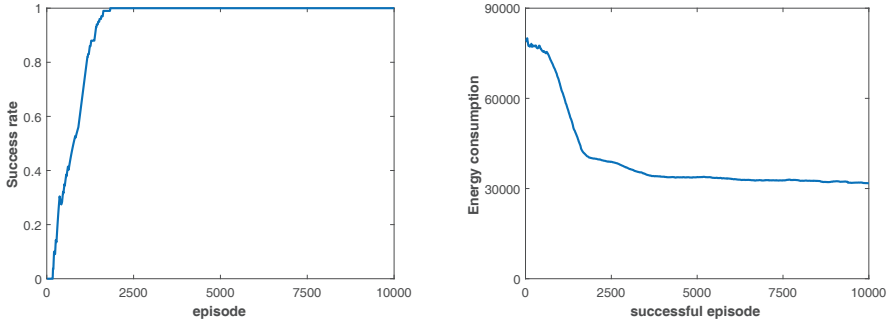
The results for various wind velocities ($1, 4, 7$ kn) and wind directions ($0°$, $90°$, $180°$) are illustrated in Figs. 6, 7 and 8 that give the performance of the proposed methodology. In particular, the progress of (a) the mean success rate of the method, i.e. the frequency of finding the target is shown, as well as (b) the calculated mean



(0°)



(90°)



(a) (180°) (b)

Fig. 6. Case 1: Results with wind velocity = 1 kn and various wind directions. Learning curves showing (a) the success rate and (b) the total energy consumption at each successful episode.

total energy consumption of the last 50 (successful) episodes. As shown in these figures, the RL agent converges to a sub-optimal solution after (approximately) 200 episodes in all cases. Naturally, the energy consumption is minimized when the wind direction coincides (0°) with the desired direction of the platform. On the

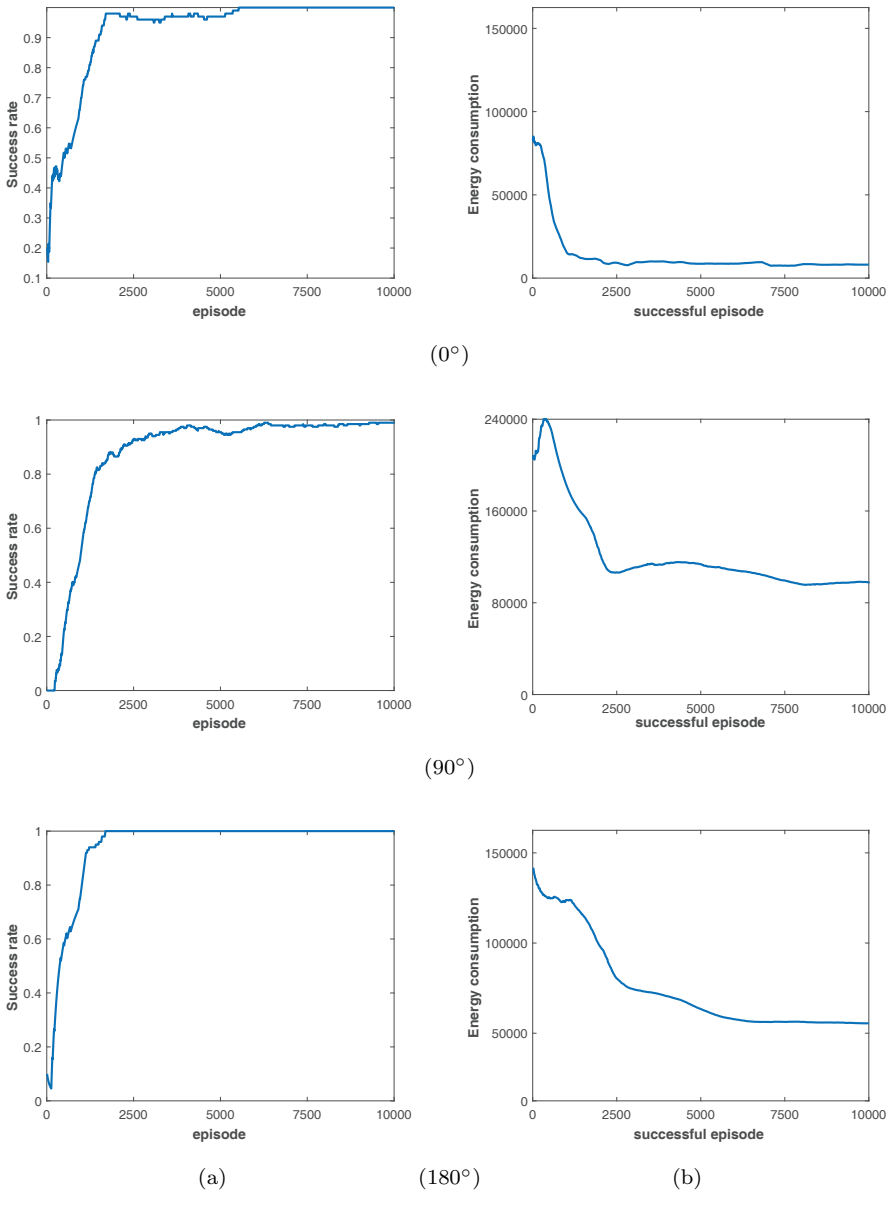
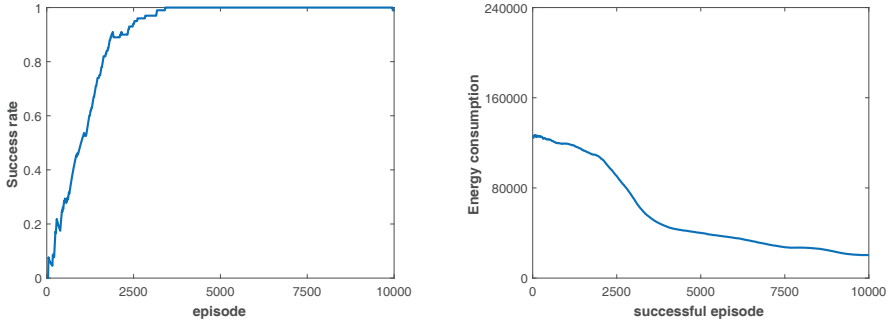


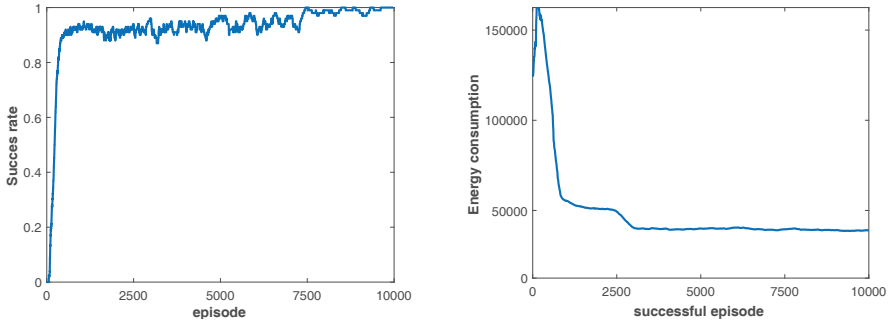
Fig. 7. Case 2: Results with wind velocity = 4 kn and various wind directions. Learning curves showing (a) the success rate and (b) the total energy consumption at each successful episode.

contrary, the energy consumption is maximized when the wind opposes (180°) the desired direction of the platform.

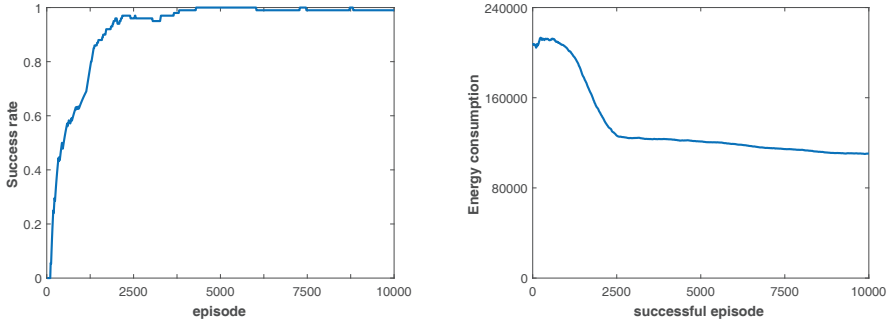
Various successful navigation paths are illustrated in Fig. 9 in the case of wind directions 0° and 180°, respectively, and wind velocity magnitude equal to 4 kn.



(0°)



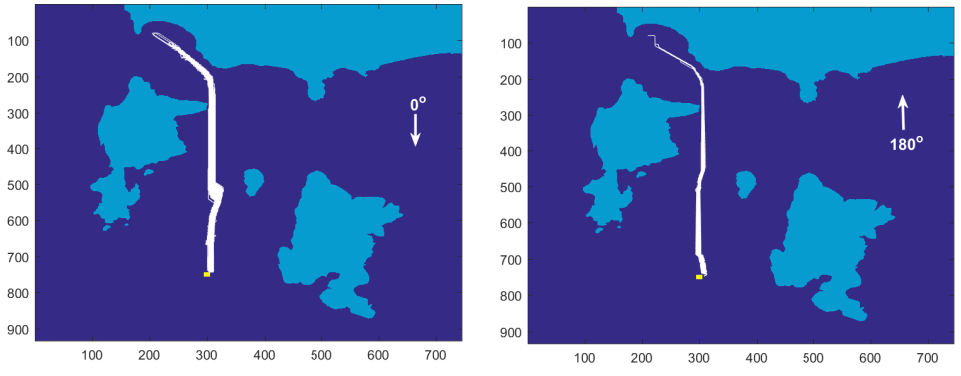
(90°)



(a) (180°) (b)

Fig. 8. Case 3: Results with wind velocity = 7 kn and various wind directions. Learning curves showing (a) the success rate and (b) the total energy consumption at each successful episode.

Apparently, the marine platform is able to discover the target point following a sub-optimal path. In addition, Fig. 10 illustrates the energy consumption and the action sequence of a sample successful episode commanded by the policy of the RL agent. As expected, the energy consumption is maximized when a change in the direction of the platform velocity is commanded by the RL agent.



(a) (b)

Fig. 9. The sub-optimal navigation paths as estimated by the proposed method in two cases of wind direction (a) 0° and (b) 180°, while the wind velocity was 4 kn.

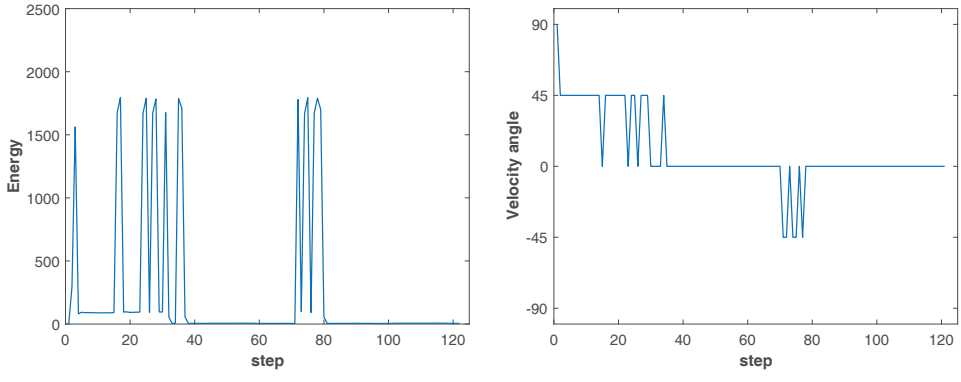


Fig. 10. Energy consumption and action sequence during a sample successful episode.

4.1. *Comparison with a baseline approach*

As mentioned before, for validation purposes we compared the proposed methodology with a baseline approach. More specific, in the baseline approach we consider the target position and the physical obstacles in the area (islets) *completely known*, in contrast to the proposed approach where the map is *unknown*. Using a path planning algorithm based on virtual potential fields, we designed an optimal path from the start position to the target position, avoiding the physical obstacles.[27] Note that the action space in the baseline approach is continuous, in contrast to the proposed framework where we consider a discrete number of 5 actions $\mathcal{A} = \{-90°, -45°, 0°, 45°, 90°\}$. For comparison reasons, the marine platform follows the desired path under the same environmental disturbances, and GPS measurement noise (see subsection 2.2.3), as in the proposed methodology. In addition, the control of the platform's actuation system is implemented using the same

velocity controller, according to (15), and (16) in subsection 2.2.4. The objective of the controller remains the same as in the proposed framework, i.e. (a) to follow the desired path with a predefined constant velocity magnitude, and (b) the desired orientation of the platform to coincide with the direction of the wind.

Figure 11 shows the desired (red line), and the simulated paths (white line) for wind velocities equal to 4 and 7 kn. In both cases, the direction of the wind velocity is the worst case scenario, i.e. opposite to the desired motion (equal to 180°).

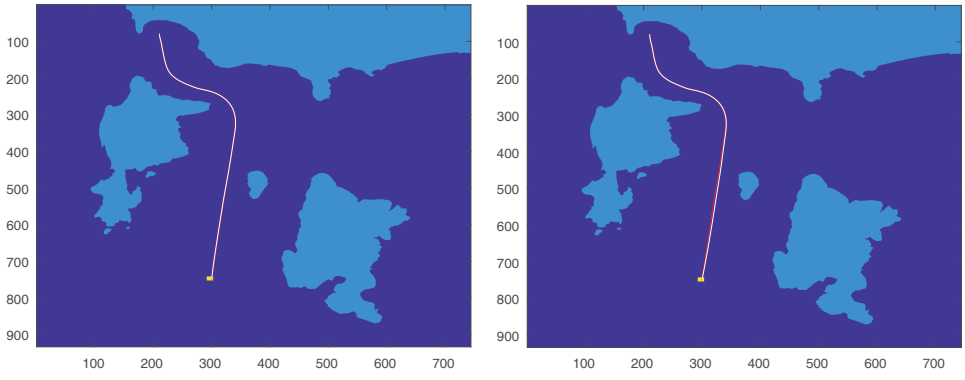


Fig. 11. The optimum path (red line), and the simulated path (white line) in the baseline approach during a sample run for (a) 4 kn, and (b) 7 kn.

Table 1 presents a comparison of the simulation results, concerning the average energy consumption per episode, the average number of steps and the average energy per step, in the baseline and the proposed approaches. As shown, the energy consumption in the proposed approach is about 25% more compared to that of the baseline approach. It must be noted that the difference in the energy consumption is mainly due to the increase of the average number of steps in the proposed framework. This behavior is explained by the use of discrete action space that may result in irregular motion and thus in additional acceleration and energy demands, especially against large wind disturbances.

Table 1. Comparison results with the baseline approach concerning energy consumption and number of steps.

	Average Energy in [kJ]		Average Number of Steps		Average Energy Per Step	
	4 kn	7 kn	4 kn	7 kn	4 kn	7 kn
Baseline	51 545	62 752	65	65	8.04	9.68
Proposed	58 361	89 152	83	94	7.03	9.48

5. Conclusions and Future Work

In this study we presented an autonomous navigation framework of a triangular over-actuated floating marine platform involving a reinforcement learning scheme, and a low-level velocity/orientation controller. The method tries to learn the policy function and estimate a target position according to a least-squares mechanism that uses RBF kernel functions. In order to achieve a sub-optimal policy with minimum energy consumption, the proposed *variable* reward depends on the energy consumption at each step. The low-level controller successfully controls the magnitude of the linear velocity equal to a predefined value, and the orientation of the platform equal to the direction of the wind resulting in reduced disturbances. Simulated results illustrated its performance under wind, and wind generated wave disturbances, and position measurement noise.

In the future, we plan to verify the proposed algorithm in a more complex simulated environment, where there is a local differentiation of the environmental disturbances. Also, an interesting direction would be to study alternative schemes of the reward function that combine energy consumption and optimal path. Moreover, including the orientation of the marine platform to the state space of the agent constitutes an interesting future research study. Another issue concerning the reinforcement learning scheme is to use alternative model-based value function approximation methodologies and to study the potential benefits of the combination of RL and Deep Learning.[28,29] Finally, the optimization of the energy consumption in an over-actuated autonomous platform, is a very interesting research topic and subject of our current research work.

References

1. M. Seto, *Marine Robot Autonomy* (Spinger-Verlag, 2013).
2. G. Antonelli, *Underwater Robots*, Springer Tracts in Advanced Robotics, Vol. 96 (Springer, 2014).
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 (1994) 185–202.
6. Y. Engel, S. Mannor and R. Meir, Reinforcement learning with Gaussian process, in *Int. Conf. on Machine Learning* (2005), pp. 201–208.
7. N. Tziortziotis and K. Blekas, Model-based reinforcement learning using online clustering, in *IEEE Int. Conf. on Tools with Artificial Intelligence* (*ICTAI*) (2012), pp. 712–718.
8. M. Strens, A bayesian framework for reinforcement learning, in *Int. Conf. on Machine Learning* (*ICML*) (2000), pp. 943–950.
9. N. Tziortziotis, C. Dimitrakakis and K. Blekas, Linear Bayesian reinforcement learning, in *Int. Joint Conf. on Artificial Intelligence* (*IJCAI*) (2013), pp. 1721–1728.

10. N. Tziortziotis, C. Dimitrakakis and K. Blekas, Cover tree Bayesian reinforcement learning, *Journal of Machine Learning Research* **15** (2014) 2313–2335.
11. B. Yoo and J. Kim, Path optimization for marine vehicles in ocean ccurrents using reinforcement learning, *Journal of Marine Science and Technology* (2015) 1–10.
12. 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* **30** (2005) 416–427.
13. 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 Int. Conf. on Intelligent Robots and Systems* (*IROS*) (2005), pp. 996–1002.
14. K. Vlachos and E. Papadopoulos, Modeling and control of a novel over-actuated marine floating platform, *Ocean Engineering* **98**(Supplement C) (2015) 10–22.
15. A. Tsopelakos, K. Vlachos and E. Papadopoulos, Design of a linear model predictive controller for an overactuated triangular floating platform, in *2014 IEEE Conf. on Control Applications* (*CCA*) (October 2014), pp. 891–896.
16. A. Tsopelakos, K. Vlachos and E. Papadopoulos, Backstepping control with energy reduction for an over-actuated marine platform, in *2015 IEEE Int. Conf. on Robotics and Automation* (*ICRA*) (May 2015), pp. 553–558.
17. M. G. Lagoudakis and R. Parr, Least-squares policy iteration, *Journal of Machine Learning Research* **4** (2003) 1107–1149.
18. L. Busoniu, D. Ernst, B. D. Schutter and R. Babuska, Online least-squares policy iteration for reinforcement learning control, in *American Control Conf.* (*ACC*) (2010), pp. 486–491.
19. K. Tziortziotis, N. Tziortziotis, K. Vlachos and K. Blekas, Autonomous navigation of an over-actuated marine platform using reinforcement learning, in *Proc. of the 9th Hellenic Conf. on Artificial Intelligence* (*SETN 16*) (ACM, New York, NY, USA, 2016), pp. 11:1–11:7.
20. K. Tziortziotis, K. Vlachos and K. Blekas, Reinforcement learning-based motion planning of a triangular floating platform under environmental disturbances, in *2016 24th Mediterranean Conf. on Control and Automation* (*MED*) (June 2016), pp. 1014–1019.
21. Nestor institute for astroparticle physics, http://www.inp.demokritos.gr/nestor/.
22. Delta Berenike, http://www.inp.demokritos.gr/nestor/dberenike/.
23. S. Hoerner, *Fluid-Dynamic Drag* (Hoerner Publications, 1965).
24. T. Fossen, *Guidance and Control of Ocean Vehicles* (John Wiley and Sons, 1994).
25. L. Li, M. Littman and C. Mansley, Online exploration in least-squares policy iteration, in *Int. Conf. on Autonomous Agents and Multiagent Systems* (*AAMAS*) (2009), pp. 733–739.
26. C. Watkins and P. Dayan, Q-learning, *Machine Learning* **8**(3) (1992) 279–292.
27. O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in *1985 IEEE Int. Conf. on Robotics and Automation* (1985), pp. 500–505.
28. Y. LeCun, Y. Bengio and G. Hinton, Deep learning, *Nature* **521**(7553) (2015) 436–444.
29. S. Gutstein, O. Fuentes and E. Freudenthai, Knowledge transfer in deep convolutional neural nets, *International Journal on Artificial Intelligence Tools* **17**(3) (2008) 555–567.