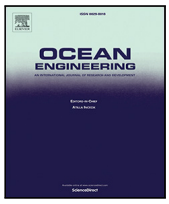
Ocean Engineering 316 (2025) 119907

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng



Research paper

A COLREGs compliance reinforcement learning approach for USV manoeuvring in track-following and collision avoidance problems

Valentin Sonntag, Adolfo Perrusquía *, Antonios Tsourdos, Weisi Guo

Faculty of Engineering and Applied Sciences, Cranfield University, Bedford MK43 0AL, UK

ARTICLE INFO

Keywords:
USV
COLREGs
Reinforcement learning
Reward design
Smoothing decision feature

ABSTRACT

The development of new technologies for autonomous platforms has allowed their integration into sea mine countermeasures. This has allowed to remove the personnel from the potential danger by having the mine search task performed by an unmanned surface vessel (USV). Traditional intelligent systems are built by agglomerating hand-coded behaviours that determine how a good manoeuvre looks like. This induces cognitive bias into the pre-defined behaviours that can violate safety and regulatory rules imposed by the COLREGs. To alleviate this issue, this paper proposes a COLREGs compliant reinforcement learning (RL) approach that gives a solution for the autonomous navigation of USVs. A custom simulation environment is developed. The RL agents are trained to deal with path-following problem with obstacle avoidance capabilities. A custom reward function is defined to consider the turning disks for the agent's decision process. A smoothing decision feature is used to smooth the transitions between consecutive actions. The results demonstrate good convergence and high performance under different scenarios. The collision avoidance with COLREGs compliances shows the effectiveness of the proposed approach under several scenarios with static and moving obstacles.

1. Introduction

Research on navigation of autonomous platforms has increased in the last years due to the proliferation of cheaper technologies that open diverse research directions and applications (El Debeiki et al., 2024; Guo et al., 2023). Critical challenges has been observed in unmanned surface vessels (USVs) due to the restricted manoeuvrability, human factors, and changes in the environmental forces such as wind, current, or waves. Here, the maritime transport is the dominant mode for the transport of goods, rising to 10.7 billion of tons in 2020 (Huang et al., 2023).

A particular research direction is to determine the navigation trajectory of USVs subject to the aforementioned constraints and objects that block the USV to follow the defined navigation path. This requires the application of different manoeuvres for collision avoidance of both static and dynamic obstacles (e.g., other boats). Here, the manoeuvres must align with the navigation rules stipulated in the International Regulations for Preventing Collisions at Sea (Organization, 2003) (COLREGs) to ensure safe navigation and protection of goods and objects in the sea (including other boats, buoys, forts, etc.).

Traditionally, USVs are operated by humans which have faced serious collision incidents due to diverse factors that include fatigue, lack of experience in the control of USVs or the use of its devices, and the misuse of the platforms by not following the safety and regulatory rules of the COLREGs (Felski and Zwolak, 2020; Lyu et al., 2024). There exist a flourishing research in how we can provide a high-level of autonomy to USVs to overcome the human factor challenges (Kong et al., 2024), whilst ensuring a safe and trustworthy performance of the USV. The algorithmic tools used in these research studies cover a wide range of methods (Vagale et al., 2021) ranging from mathematical optimization to evolutionary algorithms and artificial intelligence (AI) methods based on reinforcement learning (RL) (Busoniu et al., 2017; Yu and Perrusquia, 2021; Perrusquía et al., 2021).

1.1. Related work

Most of the works in the literature largely focused on path planning problems. Here, the autonomous system must either determine the trajectory it needs to follow to reach a desired target point, or to follow a pre-defined desired trajectory. There are plenty methods to determine the trajectory path to reach a desired target point in an optimized mechanism that includes navigable and non-navigable areas in the environment. Classical algorithms used in the literature are the Floyd (Lyu et al., 2021), Douglas–Peucker, or Land Attended Rammer–Douglas–Peucker methods (Saalfeld, 1999; Zare et al., 2021), which have been currently replaced with deep reinforcement learning (DRL)

* Corresponding author.
E-mail address: adolfo.perrusquia-guzman@cranfield.ac.uk (A. Perrusquía).

https://doi.org/10.1016/j.oceaneng.2024.119907
Received 24 July 2024; Received in revised form 7 November 2024; Accepted 22 November 2024
Available online 2 December 2024
0029-8018/© 2024 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

approaches such as deep deterministic policy gradient (DDPG) (Bildik et al., 2024b). Here, the algorithm uses input images of the topography map and the target point coordinates to solve the path planning problem (Zare et al., 2021). Other methods involve the design of customized reinforcement learning algorithms (Perrusquía et al., 2024) such as sample-efficient probabilistic model predictive control based on Gaussian processes predictions (Cui et al., 2021). This approach enables a complete model-free application that adapts to different USV characteristics.

One of the most famous algorithms used in 2D global path planning is known as Dijkstra algorithm (Luo et al., 2020). The basic idea consists in mapping the environment into a graph, and select the nodes that has the edge with shortest distance (Kang et al., 2008). This algorithm suffers of computational issues when the environment graph is too large. Other approaches are based on the A^* algorithm (Tang et al., 2021) that has the benefits of the Dijkstra algorithm combined with heuristics. Here, the environment is divided into convex zones that contains the initial and target points, such that a near-optimal path can be obtained effectively. In contrast to Dijkstra algorithm, the A^* requires less computation since it does not have to cover all nodes in the graph. However, $A*$ is commonly used in fixed environments that do not change over time, hence cannot be used for USVs under dynamic obstacles.

The path following task is achieved by determining the required actions to track the desired path as accurate as possible. Deep Q-Networks have been effectively used to manage the path following task with robust properties against environmental forces and compliance with COLREGs rules under different vessels navigating on the sea (Amendola et al., 2020). The basic idea is to use a prioritization mechanism in three specific cases: (1) if two vessels are facing each other then both of them need to execute a collision avoidance manoeuvre by turning the starboard at the right of the vessel's reference. (2) in a crossing situation, the vessel that has the other vessel on its starboard side has to give way, whilst the other vessel keeps its trajectory. Here, the area at the rear of a vessel is not applicable for the COLREGs rules since it is often a blind spot; and (3) if a vessel is coming behind another, it can overtake it on either its starboard or port side. However, the rules applied to an USV would suppose for it to give way to less manoeuvrable vessels, and in this situation that would mean almost all of the vessels according to the size of the current USV.

The most popular methods for collision avoidance are available with different data source and most of them rely on Automatic Identification Systems data (Thombre et al., 2020) and mathematical analysis methods such as Bezier curves (Li et al., 2019) or velocity analysis (Shaobo et al., 2020). In Yang et al. (2024) a human-like collision avoidance method for USVs is proposed which combines a DDQN network with an expert navigation impact factor based on fuzzy sets. Policy gradient reinforcement learning models, such as proximal policy optimization (PPO) (Bildik et al., 2024a), have been used to perform collision avoidance with COLREGs compliance with diverse source data (Meyer et al., 2020; Zhao and Roh, 2019). Here, the collision avoidance is based on either general assessment of the situation around the vessel or evaluating the risk and performing the collision avoidance for only one encounter at a time. Some strategies were applied to increase the robustness of the learning process and the adaptation of the solution to a wide range of USVs and environments (Zare et al., 2021). In this setting, a robust environmental disturbances are considered in Esfahani and Szlapczynski (2021) with an actor-critic time-delay control algorithm that enables to control the trajectory while having perturbations by the wind, waves and sea current. Recent approaches, have used deep reinforcement learning combined with other algorithms to improve specific manoeuvres of the USV. For instance, in Sun et al. (2023) design a collision avoidance algorithm based on a dynamic window approach (DWA) to transform the collision avoidance problem into a constrained optimization problem in the velocity space. Also, in Li et al. (2021) developed a path planning strategy with COLREGs collision avoidance based on a modified DQN with artificial potential field.

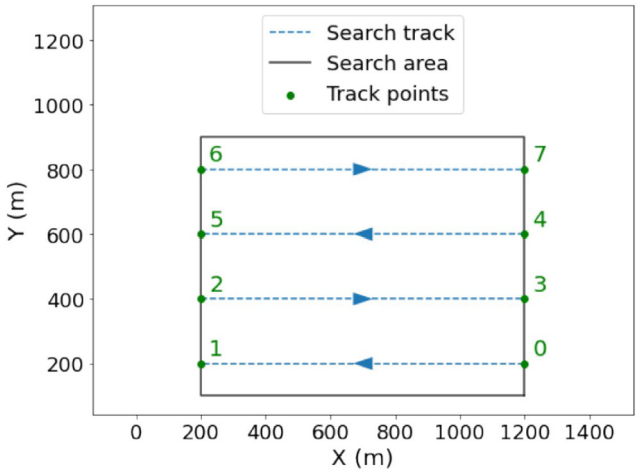


Fig. 1. Example of a track describing a research area.

1.2. Contributions

In view of the above there is a flourishing community dealing with the design of USVs that comply with COLREGs regulations. The approaches used in the literature are designed for independent regulations cases, that is, USVs that fulfil COLREGs for collision avoidance or USVs that fulfil COLREGs for path following tasks. However, what is missing is how to design an autonomous navigation methodology for USVs that comply with COLREGs' regulations in both path planning, path following, and collision avoidance scenarios.

To this end, this paper provides a novel autonomous USV navigation methodology for track-following problems under collision avoidance mechanisms that satisfy COLREGs' regulations. The approach considers the challenge of repeated U-turns throughout the path that require different manoeuvres depending on the constraints in the rate of turn (ROT) as well as the characteristics of the track. This additional feature is not considered in conventional trajectory following problems. Here, the core contributions of this work include:

1. Design, implementation and performance evaluation of different reinforcement learning algorithms applied to a path-following problem under static and moving obstacles.
2. Design of a novel reward function compliant with COLREGs' regulations.
3. A novel decision-making mechanism for the smooth transition between actions in the USV.

2. Methodology

2.1. Simulation environment

In this work we create a simple representation of real-world scenarios using USVs for both path-following problems and collision avoidance. To this end, the sea environment is modelled as a 2D plane where the USVs can freely navigate as long as they do not collide with each other or with a static obstacle. A collision system is integrated into the environment to detect collisions between a USV and a static or dynamic obstacle. Fig. 1 shows a simple visualization of the developed environment for the path-following problem.

Each kinematic description of the USV such as position, speed, heading, width and length, are given by its AIS data. In addition, we add the ROT as an additional feature in our implementation to provide a more realistic scenario. Here, both the speed v of the USV and its ROT are assumed to be constrained by

$$\begin{cases} 0 \leq v \leq v_{\max} \\ -ROT_{\max} \leq ROT \leq ROT_{\max}, \end{cases} \tag{1}$$

Table 1
Simulation parameters.

Parameter	Value
Δt	0.2 s
$d_{threshold}$	5 m
$N_{episodes}$	75
T_{lim}	$\frac{5}{2}\frac{L_{track}}{v_{\max}}$

where $v_{\max} \in \mathbb{R}$ is a maximum speed and $ROT_{\max}$ is a maximum rate of turn. The search area is given by n waypoints described by the following lumped vector $[x^0_{track}, y^0_{track}, \ldots, x^{n-1}_{track}, y^{n-1}_{track}]$. Each (x, y) coordinate in the waypoints correspond to the position of the successive entry and exit points of the search track. The total length of the track is L_{track}. Here, the USV moves smoothly to each waypoint of the track, and each time it approaches to a waypoint is within a threshold radius of $d_{threshold}$.

A simple Euler approximation model (Sun et al., 2023) is used to describe the kinematics of each USV given by the Cartesian position coordinates in x, y and the angle heading h with a sample time of Δt. The kinematic model is given by

$$\begin{cases} x(t+\Delta t) \approx x(t) + v_x(t)\Delta t, \\ y(t+\Delta t) \approx y(t) + v_y(t)\Delta t, \\ h(t+\Delta t) \approx h(t) + ROT(t)\Delta t. \end{cases} \tag{2}$$

An episode ends when one of the following scenarios is obtained: (1) the USV collide with an obstacle, (2) the USV reaches the last waypoint of the track, (3) the simulation lasts more than the pre-defined time limit T_{lim}. $N_{episodes}$ are used to train a RL agent (which will be discussed in next sessions) on random tracks with a random initial position and heading of the USV under static and dynamic obstacles. In the case of dynamic obstacles, different characteristics are randomly set such as their speed, heading, and dimensions. In this work, the speed of the obstacles are constrained by

$$0 \leq v_{obstacle} \leq 2v_{\max}. \tag{3}$$

Table 1 summarizes the parameters used in the proposed environment.

2.2. State space

In this paper, the state space is divided in two main categories: (i) variables related to the target point, and (ii) variables related to the obstacles. Define $d_{COLREGs}$ as a COLREGs compliant radius between the USV and the closest obstacle. If there are no obstacle within the radius $d_{COLREGs}$, then the obstacle variables are set to zero. The polar coordinates are used for simplicity to model all the associated variables of the USV. The following angles, distances and headings are set relatively to the USV. Here, we use the following variables as states variables

- **Target angle** denoted as α_{target} that models the angle between the USV and a target point.
- **Target distance** denoted as d_{target} that defines the distance between the USV and a specific target point.
- **Obstacle detection** denoted as $Bool_{obstacle}$ is a boolean variable that is 0 if an obstacle is not detected, and 1 if detected.
- **Obstacle angle**: denoted as $\alpha_{obstacle}$ which defines the angle between the USV and the obstacle.
- **Obstacle distance** denoted as $d_{obstacle}$ that defines the distance between the USV and an obstacle.
- **Obstacle speed**: denoted as $v_{obstacle}$ which stands to the velocity of a moving obstacle (other USV).
- **Obstacle heading**: denoted as $h_{obstacle}$ is the heading angle of the moving obstacle (other USV).

Table 2
State variables.

State variable	Name	Normalized version range
Target angle	α_{target}	$[-1, 1]$
Target distance	d_{target}	$\mathbb{R}^+$ (1 for $d_{target} = 2R_{turn}$)
Obstacle detection	$Bool_{obstacle}$	0 or 1
Obstacle angle	$\alpha_{obstacle}$	$[-1, 1]$
Obstacle distance	$d_{obstacle}$	$\mathbb{R}^+$ (1 for $d_{obstacle} = d_{COLREGs}$)
Obstacle speed	$v_{obstacle}$	$\mathbb{R}^+$ (1 for $v_{obstacle} = v$)
Obstacle heading	$h_{obstacle}$	$[-1, 1]$

These variables are normalized to accelerate the training phase of the RL agents. Table 2 summarizes the state variables used in this paper as well as their possible values when normalized.

The values R_{turn} and $d_{COLREGs}$ are defined in future sections of this paper. Therefore, the proposed state space is continuous with one boolean variable.

2.3. Action space

The action space is defined as the agent's decision vector. We use the target speed and the ROT of the USV as actions. Both the target speed and ROT are discrete variables, i.e., we have a discrete action space. We used the following values to constraint the target speed and ROT: $v_{\max} = 6$ m/s and $150 \leq ROT_{\max} \leq 600°/\text{min}$.

2.3.1. Smooth feature

A smooth decision feature is used to smooth the transition between actions in the USV. This smooth decision feature is given by an exponentially weighted average of the target actions which enables to consider the last $N_{inertia}$ decisions before applying the action to the USV.

This increases the stability of the decision-making process by injecting an inertia physical meaning to the overall process. This approach is essential for maintaining stability in long-term predictions, contributing to reliable navigation by enhancing robustness in forecasting future positions. By introducing inertia into the decision-making process, our system anticipates and incorporates the physical inertia of the USV and while accounting for potential environmental perturbations, enhancing its ability to deliver reliable long-term navigation and stability. This smooth decision feature differs from an equally weighted average that weights equally oldest and more recent decisions, which is not realistic in this setting. Here, the proposed approach weights more the most recent decisions rather than the older ones depending on the value of a user-defined coefficient $c_{inertia} \in [0, 1]$.

The key idea is to select an action from $N_{inertia}$ previous actions that has the largest inertia in a given time-window duration. This means that the selected action has a larger effect in the USV, similar to the momentum term in neural networks. Therefore, the smoother feature is basically a memory term between actions in a given time window (Perrusquía and Guo, 2023). Here, the inertia or memory that an action has in the USV is modelled by the pre-defined coefficient $c_{inertia}$. This action helps the RL algorithm to consider the actions that accelerates learning due to their larger effect. The overall action index for a given coefficient $c_{inertia}$ is

$$index_{overall} = \arg\max \sum_{n=1}^{N_{inertia}} c^n_{inertia} \begin{bmatrix} \delta_{1,index_{t-(n-1)\Delta t}} \\ \vdots \\ \delta_{i,index_{t-(n-1)\Delta t}} \\ \vdots \\ \delta_{N_{action},index_{t-(n-1)\Delta t}} \end{bmatrix}, \tag{4}$$

where N_{action} describes the number of potential actions for the nth previous action and δ_{ij} is the Kronecker delta function that acts as a selector, setting elements of the vector to 1 for the chosen action and 0 for others. This ensures that for each action step, only the coefficient

Table 3
Parameters of the Q-learning agents.

Parameter	Value
Learning rate α	0.1
α_{target} partition	$36 \times 5°$ states
d_{target} partition	5 states delimited by$[0, \frac{R_{turn}}{2}, R_{turn}, \frac{3}{2}R_{turn}, 2R_{turn}, +\infty]$
$Bool_{obstacle}$ partition	2 states
$\alpha_{obstacle}$ partition	$36 \times 5°$ states
$d_{obstacle}$ partition	$[0, \frac{d_{COLREGS}}{4}, \frac{d_{COLREGS}}{2}, \frac{3d_{COLREGS}}{4}, d_{COLREGS}]$
$v_{obstacle}$ partition	5 states delimited by $[0, \frac{v}{2}, v, \frac{3v}{2}, 2v, +\infty]$
$h_{obstacle}$ partition	$36 \times 5°$ states

corresponding to the chosen action is considered. Here, the arg max function identifies the index of the action that has the highest total weight, indicating the action with the strongest weighted accumulated preference.

To maintain a good balance between the weighted decisions, it is proposed that the most recent third of the decisions will represent half of the cumulative weights. The inertia coefficient (see Appendix) is

$$c_{inertia} = \left(\frac{-1+\sqrt{5}}{2}\right)^{\frac{3}{N_{inertia}}}. \tag{5}$$

In this paper, the corresponding values are $N_{inertia} = 15$ and $c_{inertia} \approx 0.908$.

2.4. Reinforcement learning agents

Since the state space is continuous and the action space is discrete, then this paper proposed to use the classical Q-learning and Double Q-learning (Hasselt, 2010; Perrusquía and Yu, 2021) as baselines with an appropriate state space discretization. Deep Reinforcement Learning models such as Deep Q-learning and Double Deep Q-learning (Li, 2023; Gruffeille et al., 2024; Van Hasselt et al., 2016) are also considered to effectively deal with the continuous state–space.

2.4.1. Hyperparameters selection

The ε-greedy strategy is used for the exploration–exploitation of the agents. Here, the probability ε is selected in such a way that it decays throughout the training until it reaches a value ε_{end} which is kept fixed until the simulation ends. Thus, the initial value is set to $\varepsilon = 1$ and it is updated after each episode by

$$\varepsilon \leftarrow \varepsilon_{decay}\varepsilon \quad \text{with} \quad \varepsilon_{decay} = (\varepsilon_{end})^{\frac{\Delta t}{N_{episodes}T_{lim}}}. \tag{6}$$

In this paper, the proposed values are $\varepsilon_{end} = 0.1$ and $\varepsilon_{decay} \approx 1-6.14\times10^{-6}$. The discount factor γ is set to consider the next $N_{propagation}$ steps of the simulation in a period of $T_{propagation}$. In this case, it is defined as

$$\gamma = 1 - \frac{1}{N_{propagation}} = 1 - \frac{\Delta t}{T_{propagation}}. \tag{7}$$

To enhance robustness, the system was exposed to random perturbations in its decision-making process during training, represented by the choice of the high remaining value of ε, which is ε_{end}. This approach allowed the model to learn recovery mechanisms for potential errors, as well as to account for discrepancies between its intended actions and the actual physical outcomes. These perturbations were designed to approximate unrepresented external environmental dynamics, such as wind and waves, and enable recovery from prior suboptimal decisions, effectively training the agent to handle a variety of non-ideal conditions. This stochastic element promotes the development of a more resilient policy, as the agent encounters random deviations that simulate environmental fluctuations throughout training.

In this paper, the proposed values are $T_{propagation} = 20$ s and $\gamma = 0.99$.

Table 4
Parameters of the deep reinforcement learning agents.

Parameter	Value
Hidden layers	64 and 32 neurons
Activation function	ReLU (or sigmoid)
Optimizer	Adam
Loss	Mean Squared Error
Learning rate	0.001
Memory size	500,000
Batch size	32
T_{update}	1000

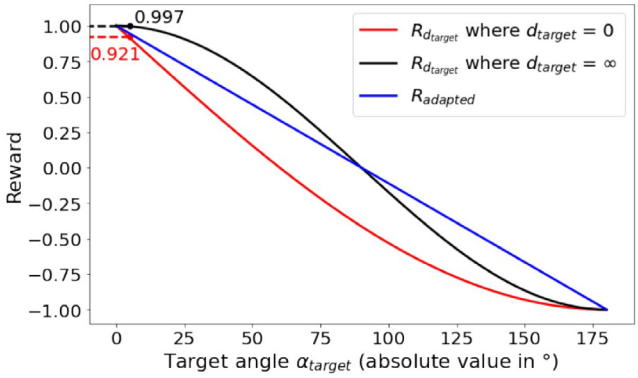


Fig. 2. Distance-based target reward (at $v = max_{speed}$).

The hyperparameters of the Q-learning and Double Q-learning agents were optimized and summarized in Table 3. Here, the proposed partitions of the state space are based on the ratios between the real states and the states domain. Table 4 summarizes the final hyperparameters for Deep Q-Learning and Double Deep Q-learning agents.

2.5. Reward function design

The proposed reward function is defined by three main elements: the reward of the target R_{target}, the reward for collision avoidance $R_{collision}$ and the reward for COLREGs compliance $R_{COLREGs}$. The values of each reward element are normalized between −1 and 1.

2.5.1. Target reward

The target reward is designed to ensure tracking of the target waypoints. We define d_{target} as the distance from the USV to the target at the time t. Here, a reward design based on this distance denoted as $R_{d_{target}}$ is given by

$$R_{d_{target}} = \frac{d^{t-\Delta t}_{target} - d^{t}_{target}}{max_{speed}\,\Delta t}. \tag{8}$$

Fig. 2 shows how $R_{d_{target}}$ changes for different angles α_{target}. Even at a distance of a few dozen metres, the reward is almost identical to the black cosine curve, representing the reward for an infinite distance between the USV and the target. The main issue is that, even with a 5° angle difference, the difference between two action rewards is not significant enough to be easily dissociated during the training, slowing it down and even decreasing the overall system performance. Indeed, when it is necessary to aim precisely at a target, the agent will miss it because when it gets close enough to have more dissociation of the different action predicted reward values, i.e., when $R_{d_{target}}$ converges to the red curve, the proximity will be such that there will be no time to correct the error and it will miss the target. This is correlated to the derivative of the reward on each point. The greater the slope, the greater the difference in the reward is obtained. The red curve representing the reward function at a null distance from the target is

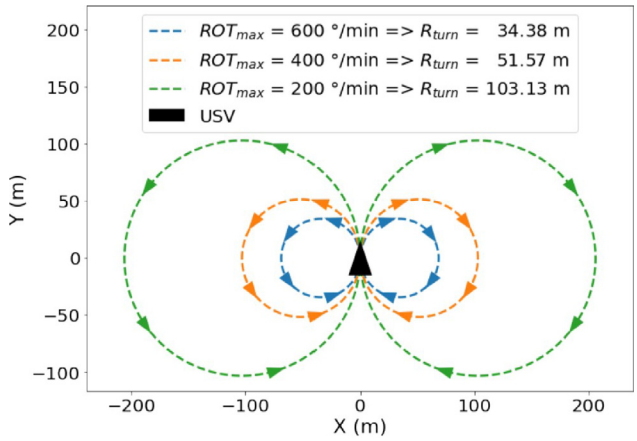


Fig. 3. Turning radius for different ROT_{max}.

better for that purpose on low angle but it still get the same issue on high angles. That is why the following selected target reward $R_{adapted}$ guarantees a constant slope on the entire angle domain

$$R_{adapted} = \left(1 - 2\left|\frac{\alpha_{target}}{\pi}\right|\right)\frac{v}{v_{max}}. \tag{9}$$

A reward based exclusively on the distance between the USV and the target leads to the agent making a decision to turn towards the target. This is a real issue when it comes to reach a target that is too close to the USV for it to reach it only in turning towards it. In fact, the ROT of the USV is not high enough for it to reach that target because there is a minimum turning radius R_{turn}, when the USV turn at its maximum ROT ROT_{max} that is limiting the region it can reach directly. As the USV can turn right or left, two regions are associated with each ROT_{max}. These regions are called the turning disks and there are delimited by their associated turning radius. The turning disks for different ROT_{max} are represented in Fig. 3.

Therefore, in the case of a target in one of the turning disks, the USV needs to perform a different manoeuvre before reaching the target point, i.e., it does not turn in the same direction as the target is detected. The target-related reward is adapted to consider this scenario and becomes

$$R_{target} = \begin{cases} \left(1 - 2\left|\frac{\alpha_{target}}{\pi}\right|\right)\frac{v}{v_{\max}} & \text{if the target is outside turning disks} \\ 1 & \text{if the target is inside a turning disk and } \alpha_{target} ROT \leq 0 \\ -1 & \text{if the target is inside a turning disk and } \alpha_{target} ROT > 0 \end{cases} \tag{10}$$

Figs. 4(a) and 4(b) respectively show how the reward R_{target} changes for different target positions around the USV.

2.5.2. Collision reward

The collision reward is designed to ensure the USV is capable to avoid collision with static and dynamic obstacles. Similarly to the target reward, we define $d^t_{obstacle}$ as the distance between the USV and the obstacle at time t, where we can design a reward in function of this distance as

$$R_{d_{obstacle}} = \frac{d^t_{obstacle} - d^{t-\Delta t}_{obstacle}}{max_{speed}\Delta t}. \tag{11}$$

Fig. 5 shows the previous reward as a function of the distance $d_{obstacle}$ and angle $\alpha_{obstacle}$. Similarly as the target reward, the difference between two actions does not produce significant reward changes that are not easily to dissociate on low angles. However, when the distance $d_{obstacle}$ decreases, the shape of the reward function is appropriate to avoid this problem as shown in the red curve of Fig. 5.

Notice that on high angles there is not a good dissociation of the reward values but it does not pose any issue since the main goal is the avoidance of the obstacle. In this case, this defines the target reward that will provide the needed difference for the agent to take the suitable decision. The collision reward is adapted to maintain this configuration regardless of the distance to the obstacle $d_{obstacle}$ as follows,

$$R_{collision} = \left(2\sin\left(\frac{\alpha_{obstacle}}{2}\right) - 1\right)\frac{v}{v_{\max}}. \tag{12}$$

Remark 1. The maritime traffic clearly defines the distance at which two ships will take collision avoidance action (Bolbot et al., 2022). However, this is can produce manoeuvres to objects that lie in the pre-defined distance threshold. Therefore, we use the angle of the target and obstacle as main features for collision avoidance. This is supported by a pre-defined COLREGs discussed in the next section.

2.5.3. COLREGs compliance reward

A COLREGs compliance radius $d_{COLREGs}$ is defined. In this paper, we propose $d_{COLREGs} = 500$ m. This reward is based on the definition of the COLREGs rules applied to a USV case. In this scenario, the USV is considered more manoeuvrable than other vessels such that the USV has to give way in all crossing situations, even if the other vessel comes from the port side. The only situation where the USV has priority is when the boat comes from the rear area of the USV, as stated in the COLREGs. In this paper, giving way results in not crossing the trajectory of the encountered vessel. This leads to the following reward,

$$R_{COLREGs} = \begin{cases} 1 & \text{if COLREGs-compliant state} \\ -1 & \text{if non-COLREGs-compliant state.} \end{cases} \tag{13}$$

The compliance regarding COLREGs rules depending on the situation is given in Table 5, where $ROT_{COLREGs}$ is a ROT describing the minimum ROT to adopt in order to conduct a compliant manoeuvre. In this paper, its value is $ROT_{COLREGs} = \frac{ROT_{\max}}{2}$.

2.5.4. Total reward

The final reward R is a linear combination of the 3 previous rewards, where the weights of the collision avoidance and COLREGs compliance are greater than the track-following since it is essential to keep the safety and regulation compliance on the top of this work priority. The final reward is given by

$$R = R_{target} + 2R_{collision} + 2R_{COLREGs}. \tag{14}$$

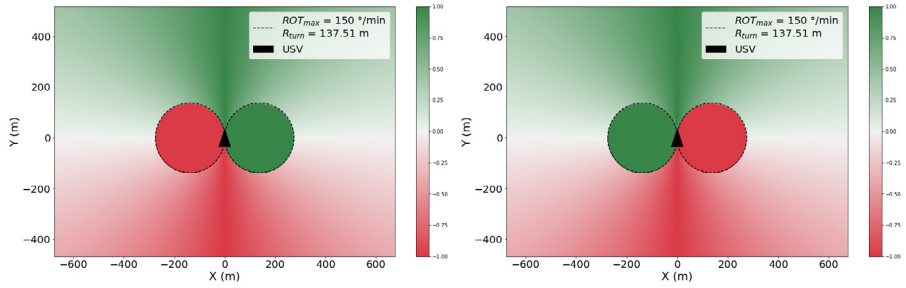
3. Results

The performance of the proposed COLREGs compliant RL algorithm is assessed under the path-following environment shown in Fig. 1. A standard laptop with Intel i7-4700HQ CPU, Nvidia GTX 765M GPU is used to train and test the proposed approach. We compare the performance of the four proposed agents (Q-learning, Double Q-learning, Deep Q-learning, and Double Deep Q-learning) under the hyperparameters of Tables 3 and 4. For more robust results, the training process is entirely iterated 10 times to provide the average, minimum and maximum of each features under study.

The score is monitored for training assessment purpose and the time taken by the USV to complete the track is used to evaluate the global performance of each agent. The score is defined as the sum of the different rewards and divided by the number of time steps per second, to which the duration of the episode is subtracted. This enables an homogenization of scores to consider the time step and penalize the duration. A time baseline is given for comparison, which is an ideal itinerary of the USV with infinite turn rate capacity and always going

Table 5
COLREGs compliance depending on the encountered situation.

Situation	State	Non-compliant action condition
No obstacle	$Bool_{obstacle} = 0$	–
Head-on	$\lvert\alpha_{obstacle}\rvert \leq 5°$ and $\lvert h_{obstacle} - \alpha_{obstacle}\rvert \geq 175°$	$ROT < ROT_{COLREGs}$
Overtaking	$\lvert\alpha_{obstacle}\rvert \leq 5°$ and $\lvert h_{obstacle} - \alpha_{obstacle}\rvert \leq 5°$	$\lvert ROT\rvert < ROT_{COLREGs}$
Crossing	$\lvert\alpha_{obstacle}\rvert \leq 5°$ and $5° < \lvert h_{obstacle} - \alpha_{obstacle}\rvert < 175°$ or $5° < \lvert\alpha_{obstacle}\rvert \leq 112.5°$	if $\alpha_{obstacle} \geq h_{obstacle} - 180\text{sign}(h_{obstacle})$ $ROT > -ROT_{COLREGs}$ else $ROT < ROT_{COLREGs}$
Obstacle abaft	$\lvert\alpha_{obstacle}\rvert > 112.5°$	–



(a) $ROT < 0$ (b) $ROT > 0$

Fig. 4. R_{target} depending on target position, with $ROT_{\max} = 150°/\text{min}$ and $v = v_{\max}$.

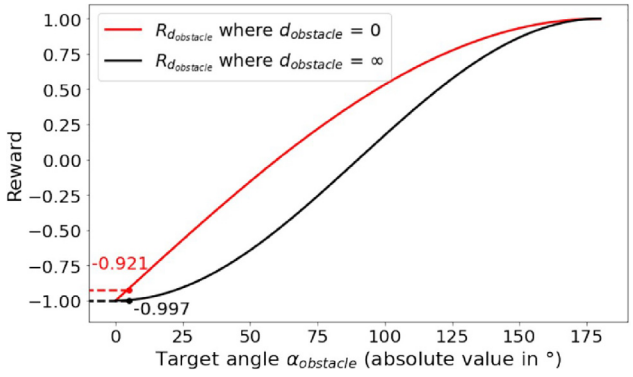


Fig. 5. Distance-based obstacle reward (at $v = max_{speed}$).

at maximum speed. In addition, a complementary comparison of the learned policy is made to control the decision process, since there is no obstacle in this scenario, so the state space has only 2 degrees of freedom and can be easily represented in a plan. Finally, the decision history is used and compared to similar versions of the system without the decision smoothing feature.

We consider two cases in our experiments: (1) when the USV can directly reach the next entry point that is the start point of the next search line, i.e. with $ROT_{\max} = 600°/\text{min}$, and (2) when the next entry point is inside a turning disk of the USV so it has to do a different manoeuvre, i.e. with $ROT_{\max} = 150°/\text{min}$.

3.1. Entry points outside turning disk

The results for a $ROT_{\max} = 600°/\text{min}$ are shown in Fig. 6. The score time-evolution in Fig. 6(a) shows that all learning processes of the agents are quite fast, i.e., less than 5 episodes for almost all of them and 100% of the agents have effectively reached the desired policy through the Q-values. This is also observed on the time to complete the track in Fig. 6(b), which converge to almost same time compared with the minimum theoretical baseline.

The DDQN agent converges in slightly more episodes compared with the DQN agent. It was expected that the DDQN agent increase the stability of the learning process and the global performance by requiring more iterations for training due to the use of two Q-networks. However, we notice that there is not a significance difference between the performances of both DQN and DDQN in this particular case study.

3.1.1. Learned trajectory

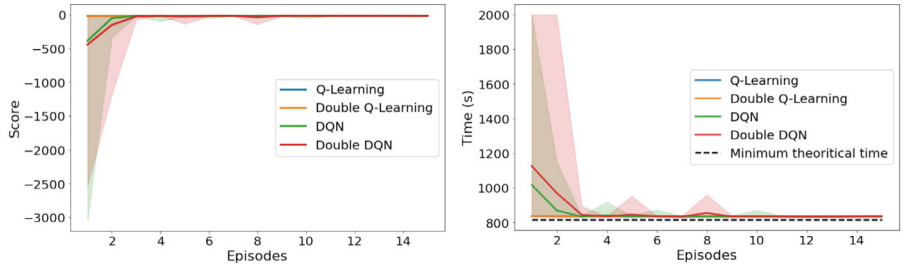
The trajectory of the USV during the learning converges towards an unique trajectory as shown in Fig. 7. The USV effectively goes straight to each target point and the trajectory respects the straight lines expected between one entry point and one exit point, such as between the points 0 and 1 of Fig. 7. However, there is a small overshoot after each entry point due to the fact that the agent targets a point and does not target a future line and its orientation, so it reaches the entry point with a non expected orientation.

3.1.2. Resulting policy

The learned policy for both DQN and DDQN is depicted in Fig. 8. Q-Learning and Double Q-Learning agents behave slightly differently if the Q-tables do not have an appropriate state space partitioning. This slightly difference is not significant for high values of $ROT_{\max}$ as in this case study, and the policy is similar to the DRL agents, with the exception of the turning disks vicinity. This area is very limited in this case because the turning radius is decreasing when $ROT_{\max}$ is increasing. Notice that the policies tend to maximize the speed of the USV.

3.1.3. Smoothing decision function

The decisions applied by the agents on the turn rate without and with smoothing decision feature are shown in Figs. 9(a) and 9(b). These results are particularly interesting because the show that with the smoothing decision feature we can obtain a smooth policy. This is expected according to the policy described in Fig. 8 and the track in



(a) Score of different agents (b) Time-to-complete-track

Fig. 6. Results with $ROT_{\max} = 600°/\text{min}$.

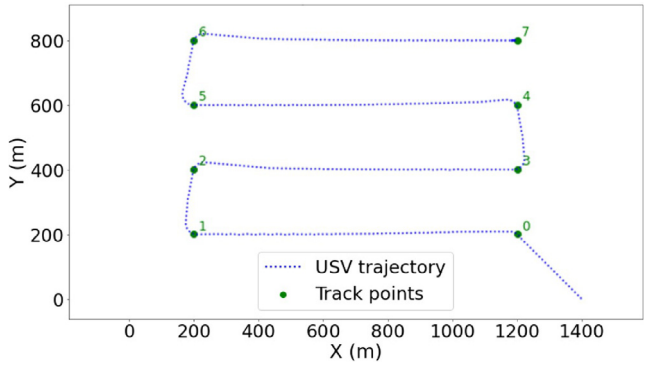


Fig. 7. Trajectory followed by the agents with $ROT_{\max} = 600°/\text{min}$.

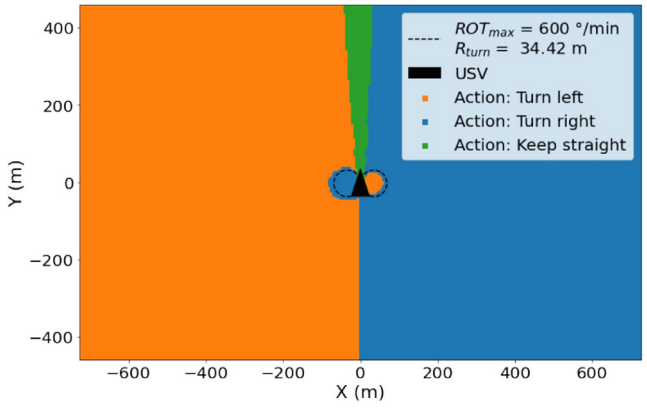


Fig. 8. Learned policy with $ROT_{\max} = 600°/\text{min}$.

Fig. 7. The straight line suggests to not turn except for some moments when there is a low angle α_{target} and when the decision becomes to turn right or left to rectify the deviation. After each entry and exit points, the instruction is to turn towards the new target if it is a entry point, or to rectify slightly the orientation deviation if it is an exit point. Without any smoothing feature, the decision is absolutely not continuous because in that case, the policy has a very small region where the decision is predicted to be not to turn so the boundary for low angle α_{target} between the turning right and left decisions is thin so the USV keeps oscillating between the two, while going straight overall. This complicates the real integration of such a system, in particular when there are some controllers used to apply the instructions to physical actuators.

3.2. Entry points inside turning disk

The training results for a $ROT_{\max} = 150°/\text{min}$ are shown in Fig. 10. The score time evolution observed in Fig. 10(a) shows that there is a learning convergence for most of the samples of each agent but some outliers still struggle to converge for the DRL agents, and are impacting the mean value of each agent's performance. This is also observed on the time needed for completing the track, which has the same similarities. However, almost all the samples reaches the near-optimal trajectory in Fig. 10(b). The difference between the time performance reached and the theoretical baseline comes from the additional time taken to undertake the manoeuvre after each exit point is consuming more time than if $ROT_{\max}$ was high enough to reach the target directly.

The outliers come essentially from a similar but not close enough estimation of the state values for a situation where the target is in a turning disk. Without the outliers, removing between 10% and 20% of the samples, and assessing the median value instead of the mean values, to be more robust regarding these outliers, the learning process is given in Figs. 11(a) and 11(b). This can be resolved by adapting the value of R_{radius} and taking some margin. In this case, the DDQN agent converges globally faster than the DQN agent, even if some outliers can cause some noise in the results.

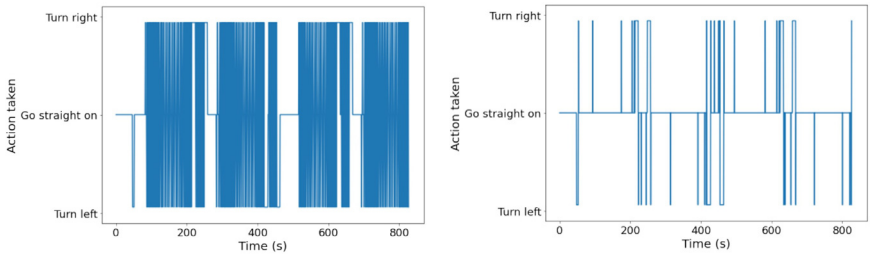
3.2.1. Learned trajectory

Fig. 12 reports the trajectory followed by the USV for all the RL agents. The USV is effectively going straight to each target point and the trajectory respects the straight lines expected between one entry point and one exit point, such as between points 0 and 1 in Fig. 12. Here, the USV performs a more complex manoeuvre rather than a higher ROT_{max}. However, there is a bigger overshoot after each entry point than for an higher $ROT_{\max}$, due to the same effect, but amplified since the ROT is lower so it takes more time to rectify the orientation difference of the USV.

3.2.2. Resulting policy

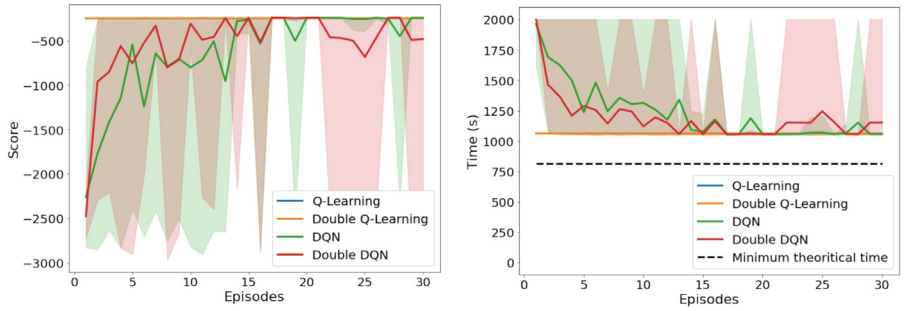
The learned policy for the DQN and DDQN is shown in Fig. 13. The policy tends to maximize the speed of the USV as in the case of a higher $ROT_{\max}$.

For low angles, the decision is to keep going straight and the others angles the decision is to turn towards the target unless it is in a turning disk. In this situation, the agent states the USV has to turn in the opposite direction, which can be observed in the trajectory of Fig. 12. However, the areas corresponding to the change of direction are not exactly matching the turning disks. The most important thing with the turning disk is to make the agent to eliminate a decision to turn towards a target inside a turning disk, so a safety margin can be used to counter this issue. For example, the right turning disk in Fig. 13 is slightly too small compared to the desired value. Setting a new value of $1.2ROT_{max}$ instead of $ROT_{\max}$ in the R_{target} reward helps increasing the safety, as well as the increase of a threshold to reduce the number of false negatives. In this case the false negatives or the situations where a



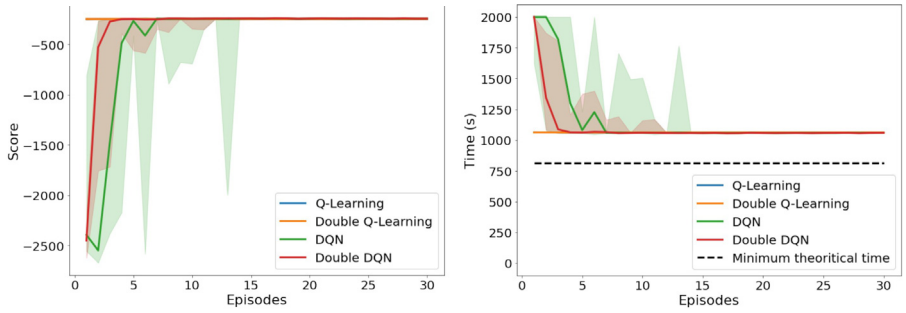
(a) Without smoothing decision feature (b) With smoothing decision feature

Fig. 9. Decision history for an episode with $ROT_{\max} = 600°/\text{min}$.



(a) Score of different agents (b) Time-to-complete the track

Fig. 10. Results with $ROT_{\max} = 150°/\text{min}$.



(a) Score of different agents (b) Time-to-complete the track

Fig. 11. Results with $ROT_{\max} = 150°/\text{min}$ and 20% margin on R_{turn}.

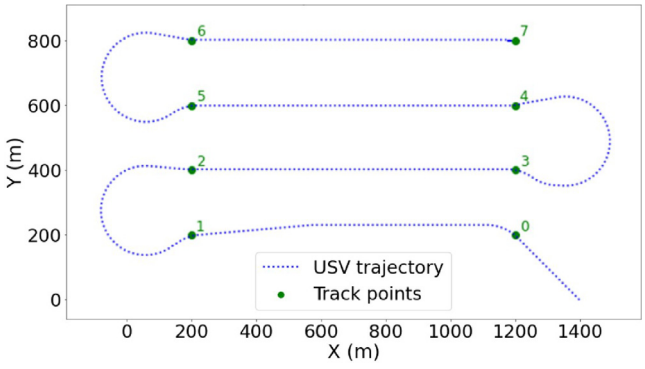


Fig. 12. Trajectory of the agents with $ROT_{\max} = 150°/\text{min}$.

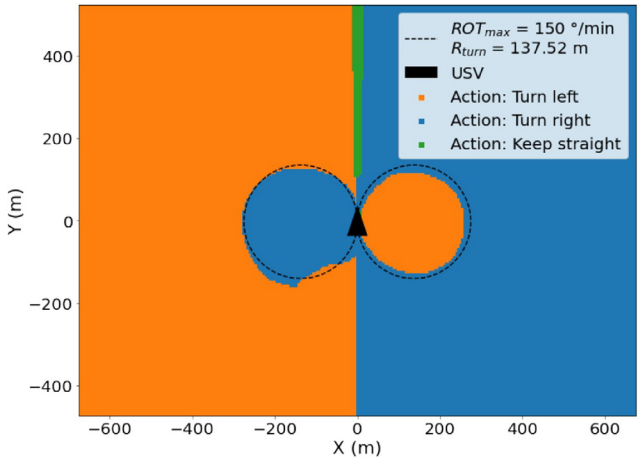


Fig. 13. Policy of DQN and DDQN with $ROT_{\max} = 150°/\text{min}$.

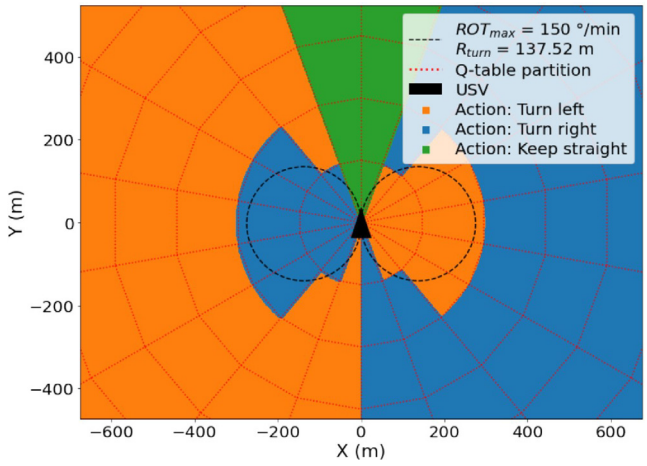


Fig. 14. Policy determined with a Q-tables with an improper partitioning.

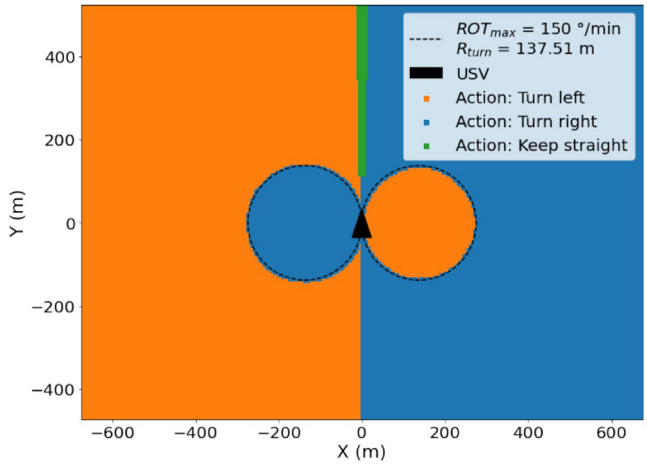


Fig. 15. Policy determined with a Q-tables with perfect partitioning.

target is inside a turning disk but it was considered to be outside. This trade-off makes the manoeuvre a little longer to execute as the USV has to move the target away from its turning disk more than necessary.

The DRL agents enables a continuous prediction for a continuous state space. However, the Q-tables of the Q-Learning and Double Q-Learning discretize the state space. For this reason, especially with a non well suited partitioning, the vicinity of the turnings disks is not reliable as shown in Fig. 14 where the partitioning is made so that a discretized state characterizes a region of 150 m × 20°.

With a better partitioning, for example the one considering the turning disks, the area in front of the USV, and the two other left and right areas, the Q-table exactly represents what are the different situations encountered for the track-following problem, as shown in Fig. 15.

The challenge is that this suitable partitioning is often inaccessible and the reason why DRL is used. Increasing the number of partitions, so the states in the Q-table, makes the training longer because of the number of different states to pass through and to update with the propagation of the reward through the steps and the update Q function.

Since the state–space variables are mostly set relatively to the USV, the learned policy benefits from enhanced generalization. In addition, the use of turning disks has the advantage of being more adaptable to the variable ROT of the USV, since disk size depends directly on the ROT. As a result, the agent effectively generalizes to unseen tracks and ROTs, as shown in Fig. 16(a) where the track simultaneously includes entry points that are inside and outside turning disks. The USV goes straight to each target point when it is possible, and performs a manoeuvre if necessary. Fig. 16(b) shows that the cumulative reward increases continuously, except when the USV stops maneuvering in the opposite direction of the next entry point and turns back towards the point. Its relative orientation causes it to move away from the targeted entry point even though it is turning towards it, leading to a negative reward. The maximum theoretical reward can be achieved if there is no need to perform a manoeuvre to reach an entry point: this limit corresponds to an infinite ROT.

3.2.3. Smoothing decision function

The decisions applied by the agents on the turn rate without and with the decision smoothing feature are shown in Figs. 17(a) and 17(b), respectively.

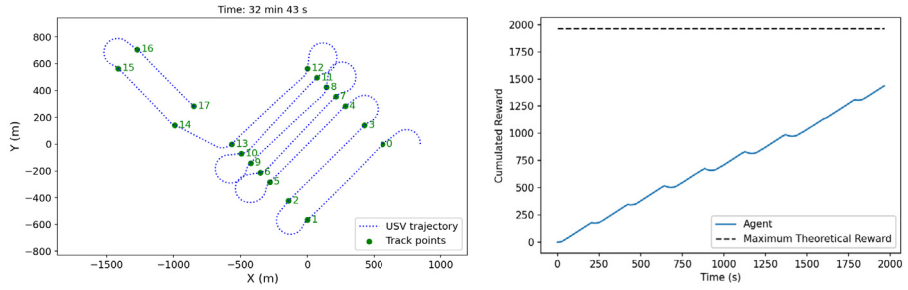
Fig. 17 shows again that, the policy is smoother as expected concerning the decisions with the smoothing decision feature, and this matches with the policy described in Fig. 15 and the track in Fig. 14. The straight line leads the USV to not turn except for some moments when there is a low angle α_{target} that keeps increasing until the boundary of the learned policy, and where the decision becomes to turn right or left to rectify the deviation. After each exit points, the instruction is to turn in the opposite way of new target to perform a manoeuvre, before coming back to it when it is outside the turning disks. After each entry points, to rectify slightly the orientation deviation. The decision is again not continuous without the smoothing decision function, with similar consequences on the policy learned.

3.3. Collision avoidance

The collision avoidance is assessed on a single line track of 2 km. It is performed by the aforementioned RL and DRL agents with the same performance: the USVs can avoid 100% of the obstacles while complying to COLREGs rules after 75 episodes. DQN and DDQN agents even reach this threshold after 50 iterations. Training the agents with more episodes seems to smooth the trajectories, whilst still performing the collision avoidance manoeuvres and complying with COLREGs rules. The training process with the collision avoidance assessment is given in Fig. 18.

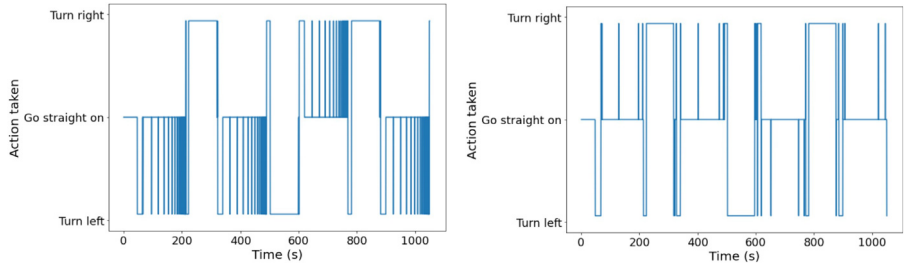
The main source of collision in the early steps is the cases when the target and the obstacle are on a similar angle perspective, i.e., $\alpha_{target} \approx \alpha_{obstacle}$. This might comes from the sum of the rewards that is too balanced to have a proper decision taken in this situation in the early training episodes. Fig. 19 shows some sample trajectories with remarkable scenarios for collision avoidance.

For the static obstacle in Fig. 19(a), the USV is avoiding the obstacle with a relative margin of around 100 m. This margin could be eventually increased to decrease the risk of collision. In the Head-On scenario in Fig. 19(b), the USV is deviating from its initial trajectory and turned right, to its starboard side, as mentioned by COLREGs rules. In a similar way as the static obstacle, a bigger safety distance could be applied to this manoeuvre. In the crossing scenario given in Figs. 19(c) and 19(d), the USV is effectively giving way to the encountering vessel when it comes from starboard side, as mentioned by COLREGs rules, and when it comes from port side as adapted to the case of an USV, seemingly more manoeuvrable. Additionally, we observe that, at the beginning of the trajectory, the USV is heading straight to the target point, before having the moving obstacle entering the area where the COLREGs is applied, relative to the designed reward. Once again, a bigger safety distance could be applied to this manoeuvre, and even a less steep trajectory. In the crossing scenario of Fig. 19(e), the beginning of the manoeuvre is similar, with the USV going in a straight line before the moving obstacle enters in the COLREGs area of effect. Then, the trajectory becomes a bit sharp but the rest is quite smooth, COLREGs compliant and even gives more distance between the USV and the other vessel than in the previous situations. The Crossing situation



(a) Trajectory of the agents (b) Cumulative reward

Fig. 16. Results for an unseen track, with $ROT_{\max} = 200°/\text{min}$.



(a) Without smoothing decision feature (b) With smoothing feature

Fig. 17. Decision history for an episode with $ROT_{\max} = 150°/\text{min}$.

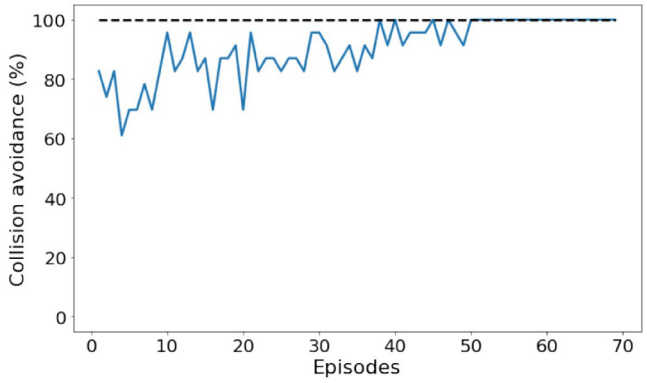


Fig. 18. Collision avoidance performance throughout the training.

from rear port side in Figs. 19(f) and 19(g) is smoother than the previous cases, and they give the possibility to assess the difference of behaviour when the speed of the incoming vessel is increasing. The manoeuvre is sharper and shorter but still in the same approach as with a slower vessel. Finally, the collision avoidance manoeuvres degrade the quality of the trajectory of the USV. Indeed, such as with the track-following problem, the target is a point and the deviation from the initial trajectory is not corrected after the collision avoidance manoeuvres. This could be beneficial in getting USV back on the desired trajectory after such an operation.

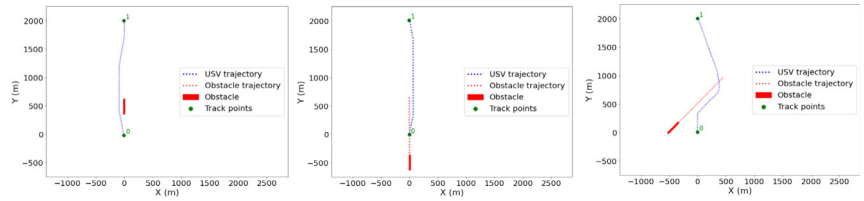
The analysis of the agent's behaviour in collision scenarios shows that it consistently adopts appropriate strategies to avoid obstacles while adhering to COLREGs requirements. The continuous collision-avoidance reward function plays a critical role here, designed specifically to encourage an increased distance between our USV and potential obstacles. This approach naturally leads the USV to avoid trajectories that head directly towards other obstacles, enabling it to achieve effective collision avoidance capabilities early in the training process. Over time, the reward function refines the decision-making process through learning, incorporating additional penalties or rewards that promote continuous adherence to COLREGs. This gradual adjustment enhances both the agent's long-term collision-avoidance precision and compliance with maritime navigation rules, allowing for more adaptive and sophisticated maneuvers in complex environments.

3.4. Computational efficiency

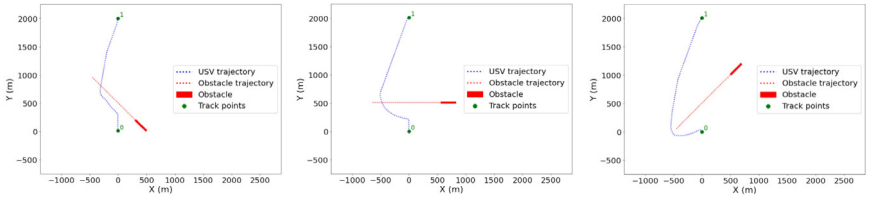
For the presented models, convergence was typically achieved within approximately 50 episodes (see Fig. 18), corresponding to between 50,000 and 250,000 steps, depending on the track complexity and environmental variability. Notably, the model's ability to follow the track emerged sooner than its collision avoidance and COLREGs compliance. This is due to the design of the reward function, which promotes quicker convergence for simpler navigational tasks such as track following only.

The training time for each step of the Q-Learning variants was around 0.4–0.5 ms and 7–8 ms for the deep Q-learning variants. Scaling to more complex environments or larger state spaces would likely require adjustments in the neural network design. The proposed deep Q-learning model is relatively lightweight, with only 2691 parameters, contributing to the rapid convergence observed in Fig. 18 compared to the typical results reported in the literature.

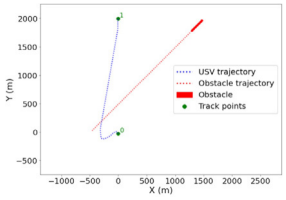
For scenarios requiring higher accuracy in environmental or USV modelling, or for more complex tracks, the model can be adapted by increasing network size or state representation granularity. This would likely extend the convergence time and raise computational costs. To support real-time deployment, further optimizations, such as model pruning, quantization, or hardware acceleration, could also be performed to reduce latency and computational resource demands.



(a) Static obstacle (b) Head-on scenario (c) Crossing-scenario from front starboard side



(d) Crossing-scenario from front port side (e) Collision scenario from port side (f) Crossing scenario from rear port side



(g) Crossing scenario from rear port sid with higher speed

Fig. 19. Collision avoidance scenarios.

4. Discussions and future work

4.1. Collision avoidance

The collision avoidance system is entirely based on one obstacle, which characteristics are encoded through 5 variables of the state vector. In case of multiple obstacles, this is the closest obstacle that is chosen as obstacle described in the state vector. In the collision avoidance assessment, only the case of a single obstacles is evaluated. It is possible to change this method by a collision risk assessment for example, or even find a way to give all the information to the agent through the state vector, the use of an image and convolutional filters could be a interesting approach to explore.

4.2. Target-related reward

The target-related reward R_{target} can be discretize somewhat like an actor-critic algorithm is using the average value of a state to define the advantage of choosing an action or another. The key idea is to have the highest difference between two similar actions to distinguish them when comes the learning process of the neural network. In practice, the method is to compute the different values of the previous R_{target} for each possible action, and return a reward of 1 for the one with the highest reward and -1 for the others. Thus it makes the neural network learn the policy rather than the Q-values, even if it virtually uses the Q-values to translate the policy, and the training is speed up and becomes more robust. However, the consequences are a loss of the expected reward of a state. It can even lead to nonsense if the discount factor γ is too high, resulting in a "bad" action that needs a lot of others to compensate physically, for example an USV missing a target, which will provide positive rewards and on the overall the sum will be higher than without doing that "mistake". More investigations can be undertaken in that direction, and especially towards policy-based reinforcement learning algorithms.

4.3. COLREGs compliance reward

By the nature of the COLREGs, the rules are not clearly stating the actions and this is free to interpret within a certain limit. In particular, there is more work particularly with some risk assessment functions, and for delimiting the area around a ship not to trespass, to enter conditionally and under increased surveillance, etc. The distance $d_{COLREGs}$ could be in this case defined more clearly and even describe a more complex area than only a disk around each vessel.

Finally, such as the state vector is only considering maximum one obstacle, the COLREGs compliance is also based on this unique obstacle and should ideally take into account all the surrounding vessels.

5. Conclusions

This paper reports a COLREGs compliant reinforcement learning approach for the safe navigation of USVs under static and dynamic obstacles. A path-following problem is used as baseline for the development of the approach. A custom reward function is designed which considers the target, the collision avoidance and the COLREGs rules. Reinforcement leaning and deep reinforcement learning agents are trained under this complementary reward under different states partitions and distance thresholds. The reported results show fast convergence and

COLREGs compliant policies that consider different scenarios given the rate of turn, the angle and the location of the target points. Several obstacle avoidance scenarios are considered under static and dynamic obstacles which effectively verify the COLREGs regulations.

Future work will consider scenarios with multiple obstacles under appropriate distance thresholds to add a new dimension of complexity to the current COLREGs rules. We hypothesize that the incorporation of attention can enable handling multiple obstacles at once while having a more adaptable and robust architecture, especially when it comes to the variable number of inputs. In addition, it can even handle multiple time-measurements per obstacle/vessel to improve overall understanding and prediction. The use of continuous actions in current deep reinforcement learning approaches such as deep deterministic policy gradient and soft actor critic are topic for further work. Also the extension to multi-agent systems is a topic that requires further investigation in this field and the incorporation of environment disturbances such as wind and waves.

CRediT authorship contribution statement

Valentin Sonntag: Visualization, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Adolfo Perrusquía:** Writing – original draft, Supervision, Conceptualization. **Antonios Tsourdos:** Supervision, Project administration, Methodology, Conceptualization. **Weisi Guo:** Writing – original draft, Resources, Project administration, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Inertia coefficient calculation

One third of the decisions represent half of the cumulative weights means in a continuous mechanism as

$$\int_0^{\frac{N_{inertia}}{3}} c_{inertia}^x dx = \frac{1}{2}\int_0^{N_{inertia}} c_{inertia}^x dx. \tag{A.1}$$

Since $c_{inertia} \in [0,1]$, we have

$$\left[\frac{c_{inertia}^x}{\ln c_{inertia}}\right]_0^{\frac{N_{inertia}}{3}} = \frac{1}{2}\left[\frac{c_{inertia}^x}{\ln c_{inertia}}\right]_0^{N_{inertia}}. \tag{A.2}$$

That means that

$$\frac{c_{inertia}^{\frac{N_{inertia}}{3}} - 1}{\ln c_{inertia}} = \frac{1}{2}\frac{c_{inertia}^{N_{inertia}-1}}{\ln c_{inertia}}. \tag{A.3}$$

Rearranging the term gives

$$c_{inertia}^{N_{inertia}} - 2c_{inertia}^{\frac{N_{inertia}}{3}} + 1 = 0. \tag{A.4}$$

Define $C = c_{inertia}^{\frac{N_{inertia}}{3}}$, we have

$$\begin{aligned} C^3 - 2C + 1 &= 0 \\ (C-1)(C^2 + C - 1) &= 0. \end{aligned} \tag{A.5}$$

Then, we have the following solutions

$$C_1 = \frac{-1-\sqrt{5}}{2}, \quad C_2 = \frac{-1+\sqrt{5}}{2}, \quad C = 1. \tag{A.6}$$

Since $c_{inertia} \in [0,1]$, so $C \in [0,1]$ such that

$$C = \frac{-1+\sqrt{5}}{2}. \tag{A.7}$$

Finally,

$$c_{inertia} = \left(\frac{-1+\sqrt{5}}{2}\right)^{\frac{3}{N_{inertia}}}. \tag{A.8}$$

References

Amendola, J., Miura, L.S., Costa, A.H.R., Cozman, F.G., Tannuri, E.A., 2020. Navigation in restricted channels under environmental conditions: Fast-time simulation by asynchronous deep reinforcement learning. IEEE Access 8, 149199–149213.

Bildik, E., Tsourdos, A., Perrusquía, A., Inalhan, G., 2024a. Decoys deployment for missile interception: A multi-agent reinforcement learning approach. Aerospace 11 (8), 684.

Bildik, E., Tsourdos, A., Perrusquía, A., Inalhan, G., 2024b. Swarm decoys deployment for missile deceive using multi-agent reinforcement learning. In: 2024 International Conference on Unmanned Aircraft Systems. ICUAS, IEEE, pp. 256–263.

Bolbot, V., Gkerekos, C., Theotokatos, G., Boulougouris, E., 2022. Automatic traffic scenarios generation for autonomous ships collision avoidance system testing. Ocean Eng. 254, 111309.

Busoniu, L., Babuska, R., De Schutter, B., Ernst, D., 2017. Reinforcement Learning and Dynamic Programming Using Function Approximators. CRC Press.

Cui, Y., Osaki, S., Matsubara, T., 2021. Autonomous boat driving system using sample-efficient model predictive control-based reinforcement learning approach. J. Field Robotics 38 (3), 331–354.

El Debeiki, M., Al-Rubaye, S., Perrusquía, A., Conrad, C., Flores-Campos, J.A., 2024. An advanced path planning and UAV relay system: Enhancing connectivity in rural environments. Future Internet 16 (3), 89.

Esfahani, H.N., Szlapczynski, R., 2021. Robust-adaptive dynamic programming-based time-delay control of autonomous ships under stochastic disturbances using an actor-critic learning algorithm. J. Mar. Sci. Technol. 26 (4), 1262–1279.

Felski, A., Zwolak, K., 2020. The ocean-going autonomous ship—Challenges and threats. J. Mar. Sci. Eng. 8 (1), 41.

Gruffeille, C., Perrusquía, A., Tsourdos, A., Guo, W., 2024. Disaster area coverage optimisation using reinforcement learning. In: 2024 International Conference on Unmanned Aircraft Systems. ICUAS, IEEE, pp. 61–67.

Guo, W., Wei, Z., Gonzalez, O., Perrusquía, A., Tsourdos, A., 2023. Control layer security: A new security paradigm for cooperative autonomous systems. IEEE Veh. Technol. Mag..

Hasselt, H., 2010. Double Q-learning. Adv. Neural Inf. Process. Syst. 23.

Huang, X., Wen, Y., Zhang, F., Han, H., Huang, Y., Sui, Z., 2023. A review on risk assessment methods for maritime transport. Ocean Eng. 279, 114577.

Kang, H.I., Lee, B., Kim, K., 2008. Path planning algorithm using the particle swarm optimization and the improved Dijkstra algorithm. In: 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application. vol. 2, IEEE, pp. 1002–1004.

Kong, X., Xing, Y., Tsourdos, A., Wang, Z., Guo, W., Perrusquia, A., Wikander, A., 2024. Explainable interface for human-autonomy teaming: A survey. arXiv preprint arXiv:2405.02583.

Li, S.E., 2023. Deep reinforcement learning. In: Reinforcement Learning for Sequential Decision and Optimal Control. Springer, pp. 365–402.

Li, G., Hildre, H.P., Zhang, H., 2019. Toward time-optimal trajectory planning for autonomous ship maneuvering in close-range encounters. IEEE J. Ocean. Eng. 45 (4), 1219–1234.

Li, L., Wu, D., Huang, Y., Yuan, Z.-M., 2021. A path planning strategy unified with a COLREGS collision avoidance function based on deep reinforcement learning and artificial potential field. Appl. Ocean Res. 113, 102759.

Luo, M., Hou, X., Yang, J., 2020. Surface optimal path planning using an extended Dijkstra algorithm. IEEE Access 8, 147827–147838.

Lyu, D., Chen, Z., Cai, Z., Piao, S., 2021. Robot path planning by leveraging the graph-encoded Floyd algorithm. Future Gener. Comput. Syst. 122, 204–208.

Lyu, H., Ma, X., Tan, G., Yin, Y., Sun, X., Zhang, L., Kang, X., Song, J., 2024. Identification of complex multi-vessel encounter scenarios and collision avoidance decision modeling for MASSs. J. Mar. Sci. Eng. 12 (8), 1289.

Meyer, E., Heiberg, A., Rasheed, A., San, O., 2020. COLREG-compliant collision avoidance for unmanned surface vehicle using deep reinforcement learning. IEEE Access 8, 165344–165364.

Organization, I.M., 2003. COLREG: Convention on the International Regulations for Preventing Collisions at Sea, 1972. IMO.

Perrusquía, A., Guo, W., 2023. Physics informed trajectory inference of a class of nonlinear systems using a closed-loop output error technique. IEEE Trans. Syst. Man Cybern..

Perrusquía, A., Yu, W., 2021. Continuous-time reinforcement learning for robust control under worst-case uncertainty. Int. J. Syst. Sci. 52 (4), 770–784.

Perrusquía, A., Yu, W., Li, X., 2021. Multi-agent reinforcement learning for redundant robot control in task-space. Int. J. Mach. Learn. Cybern. 12, 231–241.

Perrusquía, A., Zou, M., Guo, W., 2024. Explainable data-driven Q-learning control for a class of discrete-time linear autonomous systems. Inform. Sci. 121283.

Saalfeld, A., 1999. Topologically consistent line simplification with the Douglas–Peucker algorithm. Cartogr. Geogr. Inf. Sci. 26 (1), 7–18.

Shaobo, W., Yingjun, Z., Lianbo, L., 2020. A collision avoidance decision-making system for autonomous ship based on modified velocity obstacle method. Ocean Eng. 215, 107910.

Sun, X., Wang, G., Fan, Y., Mu, D., 2023. Collision avoidance control for unmanned surface vehicle with COLREGs compliance. Ocean Eng. 267, 113263.

Tang, G., Tang, C., Claramunt, C., Hu, X., Zhou, P., 2021. Geometric A-star algorithm: An improved A-star algorithm for AGV path planning in a port environment. IEEE Access 9, 59196–59210.

Thombre, S., Zhao, Z., Ramm-Schmidt, H., García, J.M.V., Malkamäki, T., Nikolskiy, S., Hammarberg, T., Nuortie, H., Bhuiyan, M.Z.H., Särkkä, S., et al., 2020. Sensors and AI techniques for situational awareness in autonomous ships: A review. IEEE Trans. Intell. Transp. Syst. 23 (1), 64–83.

Vagale, A., Bye, R.T., Oucheikh, R., Osen, O.L., Fossen, T.I., 2021. Path planning and collision avoidance for autonomous surface vehicles II: a comparative study of algorithms. J. Mar. Sci. Technol. 26 (4), 1307–1323.

Van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double q-learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 30, 1.

Yang, X., Lou, M., Hu, J., Ye, H., Zhu, Z., Shen, H., Xiang, Z., Zhang, B., 2024. A human-like collision avoidance method for USVs based on deep reinforcement learning and velocity obstacle. Expert Syst. Appl. 124388.

Yu, W., Perrusquia, A., 2021. Human-Robot Interaction Control Using Reinforcement Learning. John Wiley & Sons.

Zare, N., Brandoli, B., Sarvmaili, M., Soares, A., Matwin, S., 2021. Continuous control with deep reinforcement learning for autonomous vessels. arXiv preprint arXiv:2106.14130.

Zhao, L., Roh, M.-I., 2019. COLREGs-compliant multiship collision avoidance based on deep reinforcement learning. Ocean Eng. 191, 106436.

https://dspace.org/

A COLREGs compliance reinforcement learning approach for USV manoeuvring in track-following and collision avoidance problems

Sonntag, Valentin; Perrusquía, Adolfo; Tsourdos, Antonios; Guo, Weisi

2025-01-15
Elsevier
https://doi.org/10.1016/j.oceaneng.2024.119907
Sonntag V, Perrusquía A, Tsourdos A, Guo W. (2025) A COLREGs compliance reinforcement learning approach for USV manoeuvring in track-following and collision avoidance problems. Ocean Engineering, Volume 316, January 2025, Article number 119907

Downloaded from a DSpace repository. https://dspace.org/