X International Conference on Computational Methods in Marine Engineering
MARINE 2023
J. García-Espinosa, L. González, J.E. Gutiérrez & B. Serván-Camas (Eds)

Combining deep reinforcement learning and computational fluid dynamics for efficient navigation in turbulent flows

Artur K. Lidtke[1*], Douwe Rijpkema[1] and Bülent Düz[1]

[1] Maritime Research Institute Netherlands (MARIN) Haagsteeg 2, 6708 PM Wageningen, the Netherlands e-mail: a.lidtke@marin.nl, web page: https://www.marin.nl/en

ABSTRACT

Autonomous underwater vehicles (AUVs) face significant challenges when navigating in turbulent environments, particularly when carrying out tasks such as inspecting offshore structures that generate large turbulent wakes. These environments increase the risk of collision and damage, and decrease the success rate of recorded video frames, but carrying out such inspections with AUV offers large potential cost savings and reduced risk to human operators. Reinforcement learning (RL) combined with computational fluid dynamics (CFD) can help develop control strategies suitable for handling such complex navigation problems. The objective of this study is to assess the feasibility of such approach. To this end, two versions of the soft actor-critic algorithm are tested: one relying on the estimated vehicle position and velocity and the other augmented with pressure surface measurements obtained from fitting the vehicle with simulated pressure transducers. Both RL agents successfully navigate in turbulent flows, but the agent provided with force estimates deduced from the surface pressure has significantly improved performance. This improvement is seen in the quality of individual episodes as well as in the training robustness and speed. Therefore, this study demonstrates the potential of using RL agents to assimilate additional information for developing robust control strategies and shows the usefulness of training RL agents in high-fidelity environments, such as CFD simulations.

Keywords: Reinforcement Learning; AUV; CFD; turbulent flow.

NOMENCLATURE

(x, y, ψ)	Position and heading of the vehicle [m, m, rad]
(u, v, r)	Velocity vector [m/s, m/s, rad/s]
r	Reward function value
s	State vector
CFD	Computational Fluid Dynamics
AUV	Autonomous Underwater Vehicle
RL	Reinforcement Learning
ANN	Artificial Neural Network
ROV	Remotely Operated Vehicle
ASV	Autonomous Surface Vessel

1. INTRODUCTION

Demand for offshore structures, particularly renewable energy sources such as wind turbines and wave energy harvesters, is projected to significantly increase in the coming years and decades (Soares-Ramos

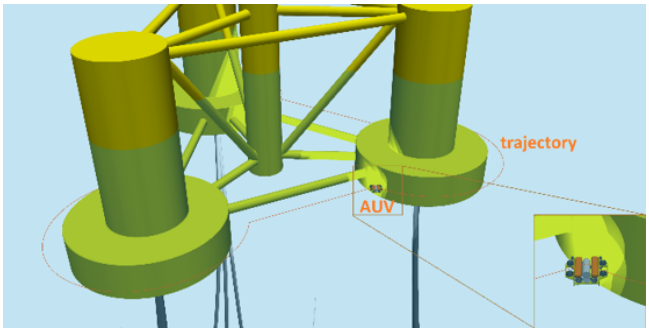


Figure 1: Visualisation of an AUV inspecting a large floating structure depicted to scale. The floater is a typical 5 MW floating wind turbine base used in the study of Burmester et al. (2020) and the vehicle is similar to a medium-sized, commercial AUV. Large difference in sizes of the vehicle and object of interest leads to large unsteady loads adversely affecting ability of the vehicle to safely navigate and carry out its task.

et al., 2020). Due to the harsh nature of the ocean environment, such structures require frequent and detailed inspections in order to guarantee their safety. Use of robotic systems, such as remotely operated or autonomous underwater vehicles, for such tasks has been common practice in the offshore industry and is making its way into the offshore wind sector (Khalid et al., 2022). However, AUVs, although potentially far more cost-effective due to their ability to operate without direct control from a crewed support vessel, are currently mostly limited only to relatively simple tasks, such as following pipelines or cables on the seabed and carrying out area surveys (Tipsuwan and Hoonsuwan, 2015). ROVs are used instead when detailed information needs to be collected about an offshore structure because of the added risk and complexity of the task and presence of strong external forces acting on the vehicle when it operates deep in the wake of a far larger structure and is also being affected by orbital velocities coming from the waves passing overhead. The difference of scales between the vehicle and the offshore structure, visualised in Figure 1, is one of the main practical reasons for this and cannot be avoided. In order to meet the expected growing demand for cost-effective offshore structure inspections, control strategies capable of dealing with this issue need to be developed.

Most AUVs in use currently rely on optimal control strategies (Tanakitkorn et al., 2018). These have been successfully applied not only in the maritime field but also across a wide spectrum of engineering problems, such as airplane autopilots. However, such methods often depend on simplified system dynamics in order to project the current state of the vehicle forward in time and thus compute the optimal control policy at any given moment within a reasonable time frame (typically tens of Hertz). When the control problem can no longer be classified as requiring disturbance rejection but rather navigation inside of a constantly present disturbance field, as would be the case for an AUV inspecting an offshore structure at short standoff distances, optimal control strategies may fail. This is because of the system dynamics becoming highly non-linear and the required frequency of control policy updates being out of reach due to the on-board computers not being able to solve the Hamilton-Jacobi-Bellman (HJB) equation quickly enough.

An alternative control approach pursued in the present work is reinforcement learning. This kind of technique has already seen use in the field of fluid dynamics, for instance, in application to active drag reduction, shape optimisation, control of renewable energy harvesters, and a plethora of general flow control problems (Viquerat et al., 2021; Rabault et al., 2020; Fan et al., 2020a; Tang et al., 2020; Vona and Lauga, 2021; Gustavsson et al., 2017; Li et al., 2021; Ren et al., 2021; Fan et al., 2020b; Anderlini et al., 2020). In the context of manoeuvring, however, their application has been more

limited. Nevertheless, several authors have demonstrated that RL can handle complex navigation and control tasks. For instance, Hong et al. (2021) used RL to control a fly-sized flyer modelled using CFD with the inclusion of the effect of wing flexibility. Several other authors in the recent literature also demonstrated control of multirotor aerial drones (Pi et al., 2021) and gliders (Novati et al., 2019). In the work of Gaudet et al. (2020), RL was also trained in order to land a descent vehicle on the surface of Mars. Here, the policy proximal optimisation algorithm was shown to be able to handle uncertainty in vehicle mass and initial position relative to the target, accept mid-flight updates to the desired goal, and maintain near fuel-optimal performance. One of the most notable applications of reinforcement learning in the maritime context is the work by Verma et al. (2018), were agents were trained in order to give fish the ability to take advantage of the energy in the flow and achieve efficient propulsion. Woo et al. (2019) also applied RL to the control problem of an autonomous surface vessel. Here, a classical guidance system based on the vector field guidance method was adopted and RL was tasked with controlling the vehicle to match the demand provided by the guidance system. The trained agent was demonstrated to successfully navigate a vessel during experimental trials. Martinsen et al. (2020) applied reinforcement learning to control an unmanned surface vessel in both simulations and trials. The results showed that the agent was capable of handling uncertain environmental factors, such as current speed, wave, and wind, while performing delicate manoeuvres in three degrees of freedom. Gunnarson et al. (2021) also demonstrated that RL allows optimal control in turbulent environments by training reinforcement learning agents to navigate in the wake of a circular cylinder by scavenging flow energy. There has also been recent progress in applying RL algorithms to time-domain simulations of manoeuvring ships. Bayezit (2022) showed promising results in identifying and accounting for complex physical interactions, such as rudder stall, using RL that could not be taken into account by classical controllers.

The main aim of the current research was to train a RL agent in order to manoeuvre an AUV inside a turbulent flow. To this end, the soft actor-critic (SAC) algorithm was coupled with the CFD code ReFRESCO[1]. The main advantage of doing so was the ability to accurately model the complex hydrodynamic behaviour of the vehicle. In parallel, a simplified training environment has been created based on a manoeuvring model of the same vehicle. This is similar to manoeuvring models typically used to test navigation and control algorithms, for example used by Martinsen et al. (2020) or Bayezit (2022), and takes far less time to evaluate compared to a CFD simulation. As such, it provided both a useful tool for testing the proposed algorithms as well as a benchmark that allowed the relative advantages and disadvantages of training RL agents in CFD to be assessed. First, the entire procedure of training the RL agent was carried out in the simplified environment. The obtained state vector, reward function, and hyperparameter choices were then used for training inside CFD. Moreover, the availability of detailed flow field information inside the CFD simulations was leveraged by providing the agent with pressure surface measurements on the surface of the vehicle. These resemble what could be achieved in reality by placing pressure sensors around the body of the AUV. The best of the agents trained with and without the local flow information were then pitched against each other, as well as the best agent trained in the simplified environment, to assess the effect of the training environment fidelity and richness of information provided in the state vector on the outcome of a challenging navigation task.

[1] `https://www.marin.nl/en/facilities-and-tools/software/refresco`

2. METHODOLOGY

2.1 The manoeuvring task

The proposed mission that the AUV was required to carry out was holding station at a specified heading in the presence of a turbulent current. Only three degrees of freedom were considered - surge, sway and yaw. This allowed the CFD simulations to be two-dimensional, greatly reducing their cost.

At the start of each episode, a random initial heading and position would be generated within a 1x1 m box domain. The vehicle was then required to navigate to the origin and point in a randomly generated direction while the bulk flow caused it to drift away and imposed unsteady loads due to the presence of turbulence in the flow. Each episode would end after a fixed number of time steps or if the vehicle moved too far from the objective. The limit was set on twice the possible initial displacement, i.e. 2x2 m around the origin.

The range of the prescribed current speeds was 0.375 to 0.625 m/s and contained free-stream turbulence with intensity of 10% (Ohshima, 2002; Orlic et al., 1992; D'Asaro and Dairiki, 1997). Integral length scale of the onset turbulence was set to 0.1 m. As this led to the turbulent eddies having a similar size to that of the vehicle, this condition was expected to exacerbate unsteady loading and make the station-keeping task particularly difficult.

2.2 Vehicle

In order to make the present computational study as realistic as possible, it was decided to base the investigated vehicle on an existing commercially available BlueROV 2 Heavy ROV. This is a relatively small remotely operated vehicle for which a wealth of information is available in the open literature, for instance in the study of Wu (2018). At the same time, in order to keep the CFD simulations cost as low as possible, it was decided not to consider the complex shape of the vehicle. Instead, a bluff body with the same overall dimensions, mass and inertia as the ROV was modelled. This led to the manoeuvring coefficients obtained from precursory CFD simulations to differ somewhat from those identified experimentally by Wu (2018). The coefficients adopted here are presented in Table 1 and Figure 2 compares an example time trace of the surge force acting on the vehicle as it is subjected to surge oscillations. It may be seen that both the simplified manoeuvring model and CFD solutions match closely.

In reality, the vehicle is fitted with four thrusters that provide actuation in the horizontal plane considered here. For this reason, controlling it requires the desired displacement in (x, y, ψ) to first be translated into a vector of generalised forces and moments along each degree of freedom. These are then translated into motor rpm set points using a thrust allocation algorithm (Tanakitkorn et al., 2018). In the present study, the last step was omitted and the generalised forces and moments were applied directly to the vehicle, thereby removing the need for modelling the actuators and simplifying the set up. It should be noted, however, that delivering a large amount of thrust leads to the formation of a large jet of water downstream of the thruster and a substantial pressure drop, both of which may affect the flow past the vehicle and thus its behaviour (Tanakitkorn et al., 2017). Including these effects in future studies would therefore be desirable. In both the CFD and simplified environments, a fixed time step of 0.02 s was used both for hydrodynamic modelling and control. Non-dimensional actions of the RL agent are obtained in the range [-1, 1]. These are scaled to the generalised forces using an assumed maximum thrust of 150 N in both surge and sway directions and maximum moment of 20 Nm.

Table 1: Key parameters of the investigated vehicle loosely based on the BlueROV 2 Heavy ROV parameters reported by Wu (2018). The presented coefficients were chosen such to ensure that the forces generated by the simplified geometry used in the CFD simulation match those of the standalone manoeuvring model.

Parameter	**Value**	**Unit**
Length, L	0.4570	m
Beam, B	0.3380	m
Mass, m	11.4000	kg
Yaw moment of inertia, I_{zz}	0.1600	kg m^2
Quadratic drag coefficient, X_{uu}	-40.1778	kg/m
Quadratic sideforce coefficient, Y_{vv}	-105.4842	kg/m
Quadratic yaw moment coefficient, N_{rr}	-1.5500	kg m^2 rad^{-2}
Linear drag coefficient, X_u	-8.9063	kg s m^{-2}
Linear sideforce coefficient, Y_v	-30.2914	kg s m^{-2}
Linear yaw moment coefficient, N_r	-0.0700	kg m^2 s^{-1} rad^{-1}

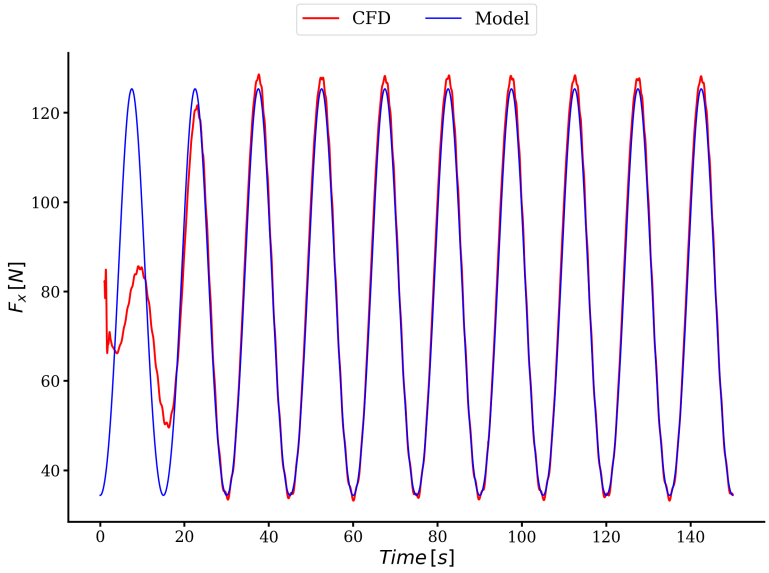


Figure 2: Unsteady surge force acting on the AUV when it is subjected to a prescribed oscillatory motion. Direct CFD results are compared to those obtained using the simplified manoeuvring model. Slow ramp-up of the motion was used at the start of the CFD computation, which is why the forces differ during the initial few motion cycles.

One of the RL agents was provided with local pressure information collected on each of the flat sides of the vehicle, as depicted in Figure 3. In order to aid the agent in assimilating this information, it was translated into a crude force estimate by multiplying the registered pressure, p, with the local normal vector, $\hat{n}$, and face area, A, leading to

$$F_i = \sum_{j=0}^{N_{sensors}} \hat{n}_{i,j} p_j A_j. \tag{1}$$

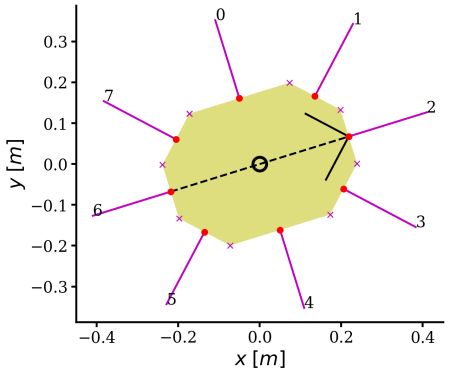


Figure 3: Placement of the pressure probes (indicated by crosses) on the simplified AUV shape, including the local surface normal vectors used to come up with force estimates that are provided to one of the RL agents.

The force estimated using Eq. (1) was then non-dimensionalised with the current magnitude of the vehicle speed available to the agent using

$$C_i = \frac{F_i}{\left(\frac{1}{2}\rho|\mathbf{U}|^2\right)}. \tag{2}$$

Figure 4 presents the result of this approximation collected during one of the training runs and compared to the ground truth provided by the CFD computations. As expected, there is a clear linear trend in the data but the estimates are far from perfect due to the local nature of the collected information. Furthermore, more variability is present in the sway direction, mostly because the blunt shape of the vehicle causing more vortex shedding and unsteady hydrodynamic force when its side is exposed to the flow.

2.3 Simplified environment

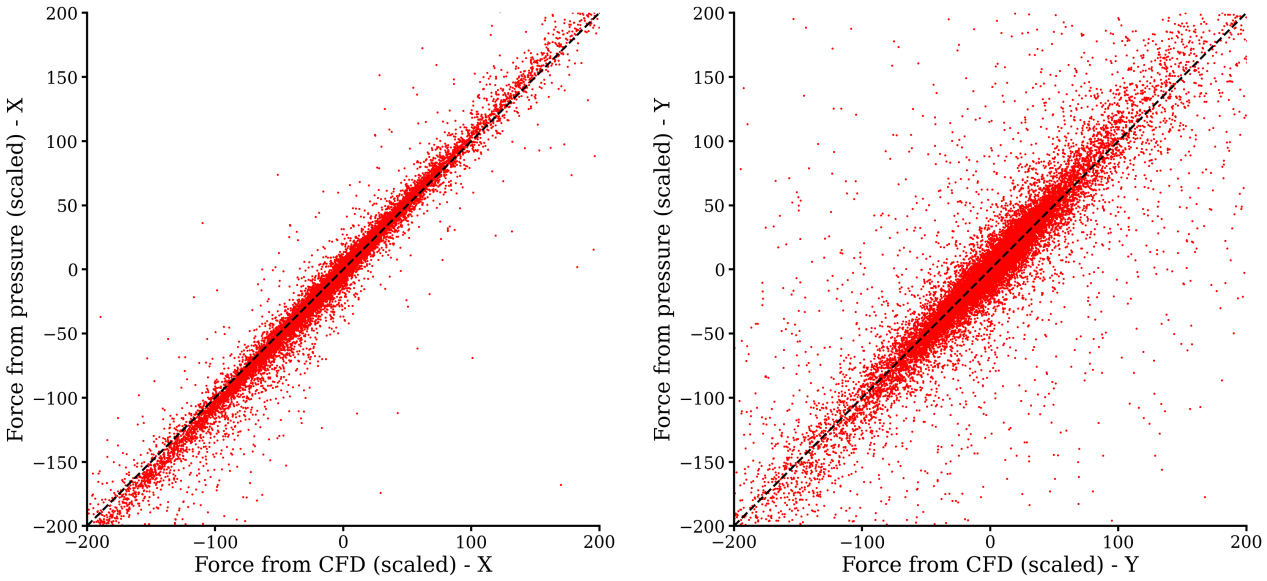
Dynamics of an underwater vehicle moving in the horizontal plane obey (Fossen, 2011; Tanakitkorn et al., 2018)

$$\mathbf{M}\dot{\nu} + \mathbf{C}(\nu)\cdot\nu + \mathbf{D}(\nu)\cdot\nu = \tau, \tag{3}$$

where the $\nu = [u, v, r]^T$ is the velocity vector, $\tau = [X, Y, N]^T$ is the generalised force vector, and the bold symbols refer to the inertia, Coriolis-centripetal, and damping matrices, respectively. These comprise of several manoeuvring coefficients and the position of the centre of gravity inside the vehicle reference frame:

$$\begin{gathered} M = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - Y_{\dot{u}} & 0 \\ 0 & 0 & I_{zz} - N_{\dot{r}} \end{bmatrix}; \quad D = \begin{bmatrix} X_u + X_{|u|u}|u| & 0 & 0 \\ 0 & Y_v + Y_{|v|v}|v| & Y_r + Y_{|r|r}|r| \\ 0 & N_v + N_{|v|v}|v| & N_r + N_{|r|r}|r| \end{bmatrix}; \\ C = \begin{bmatrix} 0 & 0 & -m(x_{cg} + v) + Y_{\dot{v}}v \\ 0 & 0 & -X_{\dot{u}}u + mu \\ m(x_{cg} + v) - Y_{\dot{v}}v & X_{\dot{u}}u - mu & 0 \end{bmatrix}. \end{gathered} \tag{4}$$

In the present study, coefficients for the BlueROV 2 Heavy presented by Wu (2018) were utilised. In this study, they were derived using laboratory experiments. The coefficients were further tuned in order to increase the similarity between the results of the manoeuvring model and the CFD simulation



(a) Axial (surge)

(b) Transverse (sway)

Figure 4: Comparison of the force instantaneous force coefficients acting on the vehicle as computed using CFD and estimated using the local pressure measurements.

of the simplified geometry. Furthermore, the added mass terms were ignored in order to make the equations of motion easier to solve. This introduces a significant inaccuracy to the model, but it should be recalled that in the present study this serves only as an interim environment used to test the RL strategy before the much more expensive training inside CFD is carried out. The coupled set of ordinary differential equations presented in Eq. (3) was advanced in time using the Euler method using the same time step size as that used for the CFD simulations (0.02 s).

Unsteady loads on the vehicle were applied by first sampling an isotropic turbulence field inside a separate CFD simulation. The data were then processed using a spectral proper orthogonal decomposition technique presented by Lario et al. (2022). This allowed the dimensionality of the data set to be reduced while being able to replay the flow. Unsteady flow velocity was incorporated into the equation of motion following the Lagrangian principle. In other words, the vehicle was assumed to move in a reference frame fixed to the local flow velocity that comprised of the uniform and fluctuating components. This approach neglects the complex hydrodynamic interactions between turbulent eddies and the body of the vehicle, but, at the same time, is affordable to compute and causes the motion of the vehicle not to be purely deterministic.

2.4 CFD environment

In order to allow arbitrary current directions, two grid regions were utilised with a sliding interface between them, as shown in Figure 5. At the start of each simulation, the inner domain is rotated such that the current approaches the vehicle from the desired heading even though the flow is always prescribed to flow in the negative x-direction. This allows the outer boundaries of the domain to remain fixed, simplifying the simulation set up and allowing well-posed boundary conditions to be used. Turbulence is introduced into the flow using a synthetic inflow turbulence generator, described in Lidtke et al. (2021). This was opted for instead of placing a physical obstacle upstream of the objective in order to reduce the cost of the simulations. The inflow turbulence generator (ITG) is

located 4 m upstream of the origin. This is far enough from the origin in order to make sure that the turbulence is fully developed while still placing the inside the rotating grid region. This serves to prevent the turbulence from being disrupted or damped by interpolation at the sliding interface.

Inside the inner grid region, grid deformation is applied in order to facilitate the motion of the AUV. This is solved using an implicit predictor-corrector method coupled to the iterative pressure-velocity solution algorithm. Deformation of the grid is computed using radial basis functions. In total, approximately 100,000 cells were used in order to discretise the domain. Non-dimensional wall spacing was chosen such that $y^+ < 1$, allowing the boundary layer to be fully resolved.

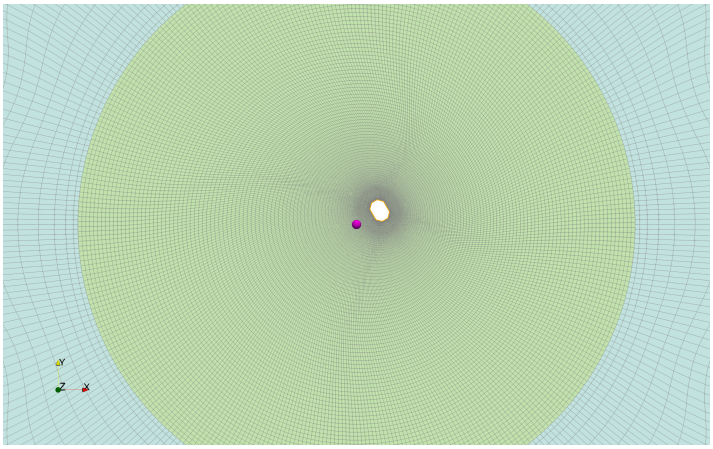


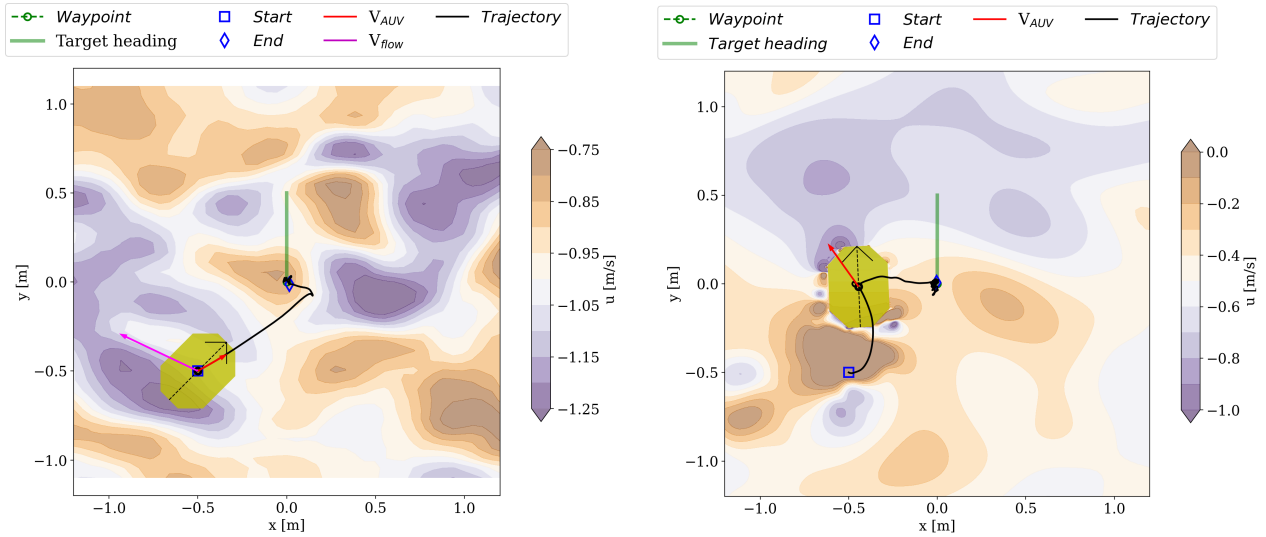
Figure 5: An overview of the CFD set up of the AUV simulation. Pink sphere denotes the origin, which also marks the objective.

The flow was solved using Partially Averaged Navier Stokes (PANS) approach based on the KSKL turbulence model (Klapwijk et al., 2021). This was necessary in order to allow the generated turbulence to convect downstream from where it was introduced into the flow and interact with the vehicle in a realistic fashion. While two-dimensional turbulence is in and of itself an artificial concept, the generated fluctuations did nevertheless obey the governing equations and were therefore deemed realistic enough for the purpose of providing strong external disturbances to the vehicle. During each simulation, the vehicle's motion was frozen for the first 750 time steps in order to allow the turbulent flow to develop. Data collected during this period was not used in the training of the RL agent.

Figure 6 compares instantaneous snapshots of both the CFD and simplified environments. The large turbulence structures may be seen to strongly interact with the vehicle. Furthermore, because of its blunt shape, the presence of the mean current itself is also sufficient to give rise to local vortex shedding and, consequently, unsteady loads affecting the vehicle dynamics. In the simplified environment all these detail are absent and the background flow only affects the motion solver in a limited way. What is more, the presence of the vehicle is assumed not to have an effect on the fluid. This assumption is demonstrably incorrect but necessary to make in manoeuvring model-based approaches.

2.5 Reinforcement Learning

As the RL agent, the soft-actor-critic (SAC) approach was utilised. This is a modern and popular RL algorithm for control in continuous state and action spaces. It is an off-policy actor-critic algorithm based on the maximum entropy reinforcement learning framework with the actor and critic represented by separate neural networks. It was designed to improve both sample efficiency and reduce sensitivity



(a) Simplified environment

(b) CFD environment

Figure 6: Example snapshots of the CFD and simplified environments. In the former, a complete description of the flow is available and the forces acting on the vehicle can be accurately determined. In the simplified case, only the fixed manoeuvring coefficients are utilised in order to determine the motion of the vehicle in a reference frame following the local flow velocity.

to hyperparameters, see Haarnoja et al. (2018a) and Haarnoja et al. (2018b) for more details. In this study, the SAC implementation from the open-source RL library Stable-Baselines3 was employed Raffin et al. (2021).

Two advantages of the SAC are that first it is an off-policy algorithm, where experiences are sampled from an (experience) replay buffer, resulting in improved sample efficiency. Second, an additional entropy term is added to the RL objective in order to simultaneously maximize the reward and exploration in regions of large uncertainty. The relative contribution of the entropy term and thus the amount of exploration in SAC is governed by a temperature parameter. In earlier versions of the algorithm, the temperature parameter was a hyperparameter manually set by the user, but in Haarnoja et al. (2018b) an automatic procedure to adapt the entropy was introduced. In this study the latter approach was adopted as implemented in the Stable-Baselines3 library.

The SAC agent also allows the use of continuous state and action spaces, which are often encountered in control problems. It was applied successfully to other relevant use cases, e.g. heading keeping if ships in waves Bayezit (2022), collision avoidance of unmanned aerial vehicles (UAVs) Lee and Moon (2022) and AUVs Xu et al. (2022), and path following control of unmanned surface vessels (USVs) Zheng et al. (2022).

In order to define the state vector, first the position and heading errors are computed as

$$\begin{aligned} \epsilon_x &= x_{goal} - x, \\ \epsilon_y &= y_{goal} - y, \\ \epsilon_\psi &= \psi_{goal} - \psi, \end{aligned} \tag{5}$$

where the demanded heading is randomly selected for each episode and the target position is at the origin. It should also be noted that in the above equation, the comparison of current and desired

heading angles needs to be carefully handled in order to wrap the angles around 0 and 2π, therefore providing the vehicle with the smallest signed error to the desired orientation. The state vector is then assembled from the following elements:

$$s = \max\left(-1,\ \min\left(1,\ \left[\underbrace{\frac{\epsilon_x^i}{0.2}, \frac{\epsilon_y^i}{0.2}, \frac{\epsilon_\psi^i}{\frac{\pi}{4}}}_{\text{Absolute error}}, \underbrace{\frac{\epsilon_x^i - \epsilon_x^{i-1}}{0.025}, \frac{\epsilon_y^i - \epsilon_y^{i-1}}{0.025}, \frac{\epsilon_\psi^i - \epsilon_\psi^{i-1}}{\frac{2^\circ\pi}{180^\circ}}}_{\text{Rate of change of error}}, \underbrace{\frac{u^i}{0.2}, \frac{v^i}{0.2}, \frac{r^i}{\frac{30^\circ\pi}{180^\circ}}}_{\text{Velocity vector}}\right]\right)\right). \tag{6}$$

In the above, superscripts i and $i-1$ refer to information available at the current and previous time step, respectively. The scaling of the position error has been chosen based on the dimensions of the vehicle as it was assumed that an error smaller than half the vehicle length can be deemed acceptable. The same held for heading errors within 45 degrees of the demanded value. Rate of change of error and velocity scaling has been selected based on preliminary results obtained using a PID controller. Note that in this context, r refers to the angular velocity and not the reward function. For the agent using pressure sensors, the scaled force estimates obtained from the pressure sensor readings described by Eq. (1) were also appended to the state vector, leading to 11 elements in total.

It is worth noting that at present it is assumed that the vehicle's position, heading, and velocity vector are known and certain. However, in practice this is often not the case (Zhou et al., 2017; Munafò et al., 2014). This is because the lack of availability of fixed reference points and GPS signals underwater leads to a steady growth of estimates available from the inertial navigation systems (INS). One way to overcome this in practice is to invest in Doppler velocity logs (DVL), which measure the vehicle speed relative to the flow, or acoustic navigation aids that provide a way to estimate the vehicle position exactly. Furthermore, estimates obtained from INS and augmented with information from additional sources, such as the DVL, is inherently noisy and needs to be carefully treated before being used for guidance and control. A typical approach is the Kalman filter, but several alternative approaches exist (Zhou et al., 2017). In the present study this layer of complexity has been ignored but one should keep it in mind when applying the presented algorithms in practice.

Another important aspect of the environment is the reward function. This has been largely inspired by the work of Woo et al. (2019) and aims to, first and foremost, drive the error in vehicle position and heading to zero by providing a positive reward when these two quantities are small. In order to avoid issues with the heading converging to a value π away from the desired value, a penalty is applied for errors larger than 90 degrees. Furthermore, in order to encourage smooth actuation, root mean square value of the most recent 20 actions is computed. This is done first for each action and then a mean value of the action vector is computed. Finally, a penalty is assigned if the vehicle exceeds the predefined domain. The reward function is illustrated mathematically in Eq. (7) and depicted in

Figure 7.

$$r = \underbrace{e^{-5\sqrt{\frac{\epsilon_x^2+\epsilon_y^2}{2}}}}_{\text{Encourage position error minimisation}} + \underbrace{\begin{cases} e^{\left(-0.1\left|\epsilon_\psi \frac{180^\circ}{\pi}\right|\right)} & |\epsilon_\psi| < \frac{\pi}{2} \\ -e^{\left[-0.1\left(180^\circ - \left|\epsilon_\psi \frac{180^\circ}{\pi}\right|\right)\right]} & \text{otherwise} \end{cases}}_{\text{Encourage heading error minimisation and discourage converging to } \pi \text{ away}} + \underbrace{+e^{-0.6A_{RMS}}}_{\text{Encourage small RMS of the most recent N actions}} + \underbrace{\begin{cases} -100 & \text{if out of bounds,} \\ 0 & \text{otherwise} \end{cases}}_{\text{Out-of-bounds penalty}}. \tag{7}$$

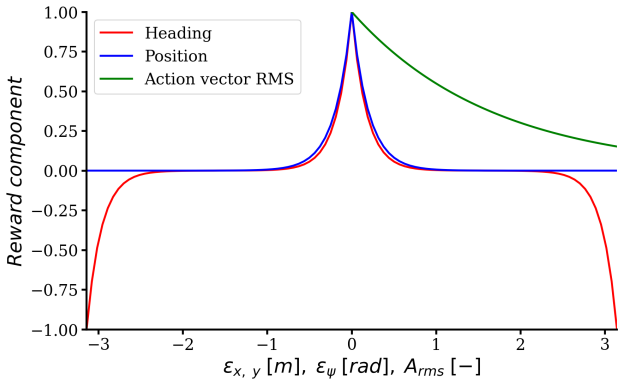


Figure 7: Illustration of the reward function components for a typical range of x-coordinate values.

Table 2 lists the hyperparameters of the SAC agent used in the simplified environment. Three different instances of the agent were trained in order to assess the variability in its performance given the stochastic nature of artificial neural network initialisation and training process.

Table 2: Hyperparameters of the RL agent

Parameter	**Value**
Learning rate	0.003
No. neurons - actor	[128, 128]
No. neurons - critic	[128, 128]
Batch size	256
Buffer size	2^{17} (131,072)
No. training steps	300,000
Discount factor (γ)	0.95

Coupling of the reinforcement learning algorithm implemented in Python with the CFD code was implemented via a TCP/IP interface. At each CFD time step, the CFD code would collect the relevant information (position, heading, velocity vector, surface pressure values), and pass it to the

Python program in a fixed-format message. The information was then decoded and used at each step of the OpenAI gym-compatible environment[2] in order to compute the reward and store the state-reward tuple in the replay buffer. The Python program would also decide when the episode should be prematurely terminated and was responsible for spawning new computations on the supercomputer cluster in order to advance the training process.

3. RESULTS

3.1 Training

Figure 8 presents the progress of training of agents inside the CFD simulations. A notable feature of the results is the significant scatter of outcomes obtained when training the agents without access to local pressure information. On the other hand, once this information is made available, the agents very quickly and robustly converge to a satisfactory result in terms of being able to stay within the domain (maximum episode length reached), achieving a high total reward (reaching and staying close to the objective), and showing consistent improvement. Interestingly, one of the simple agents did manage to reach a comparable result. It is unclear what set it apart from the remaining two that exhibited far inferior performance as inferring such differences from comparisons of artificial neural networks is a non-trivial task. The best of each category of agents were selected for future application and testing.

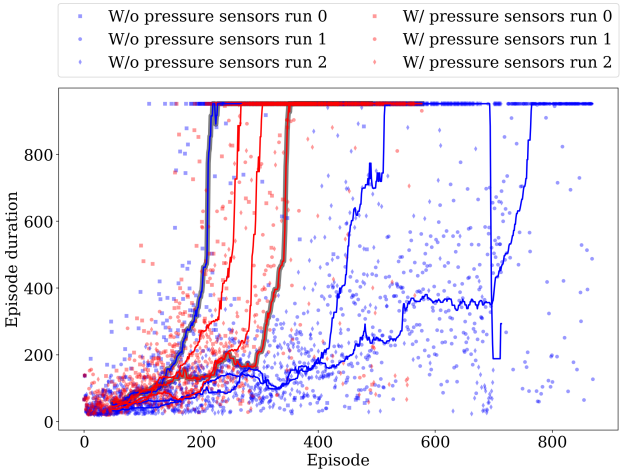
3.2 Evaluation

In order to further investigate the performance of the RL agents, they were all given the task of steering the vehicle under deliberately challenging conditions. To that end, they were required to start the episode at the furthest extent of the domain, $\mathbf{x} = (-0.5,\ -0.5)\ m$ with an initial heading of 45° to the flow and mean current velocity of 0.5 m/s. This meant both a very large surge and sway force that the agents had to initially overcome. The final heading was set to 90°, leading to a very large unsteady sway force because of the vehicle's side being square with the onset flow and a large separation zone appearing. The free-stream turbulence and associated time-varying loading further added to the difficulty. In addition to the agents trained in the CFD environment, one trained in the simplified environment was also included. Furthermore, a simple PID controller was also simulated. This had been tuned in the CFD simulations without the presence of a turbulent current.

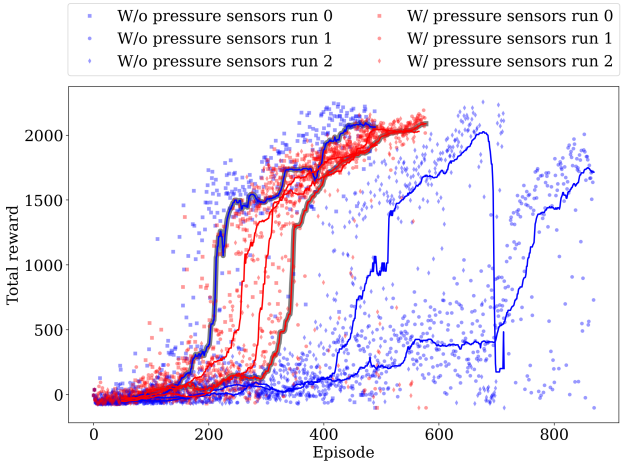
Figure 9 presents the temporal evolution of the vehicle position and heading. The RL agent trained in CFD without access to pressure information clearly exhibited the worst performance, completely missing the goal in the y-direction and showing large oscillations in the response. The agent trained in the simplified environment handled moving and staying on target appreciably well, but could not maintain a steady heading, leading to large oscillations in that degree of freedom. Conversely, the PID controller achieved satisfactory performance but struggled to maintain a steady course and position. The best results have been obtained using the agent trained inside CFD with access to surface pressure information. It managed to quickly converge to the objective with little overshoot and maintain both steady position and heading with a minimum amount of oscillations.

These qualitative comparisons may be further refined by computing the RMS error along each degree of freedom during the last 500 time steps of each simulation. These results are presented in Figure 10 and confirm the observations already discussed in the context of the time series analysis.

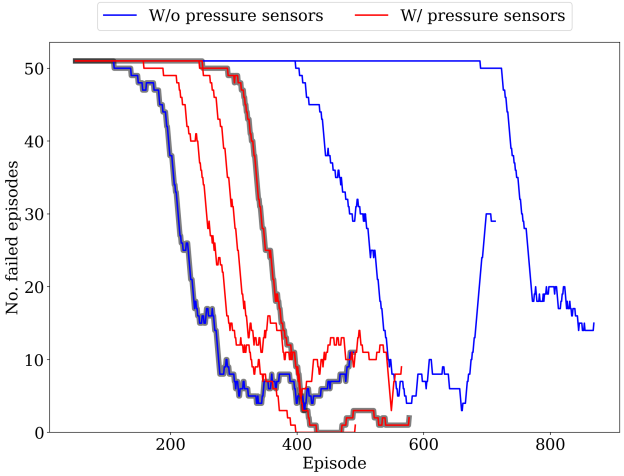
[2] `https://github.com/openai/gym`



(a) Episode duration expressed in time steps

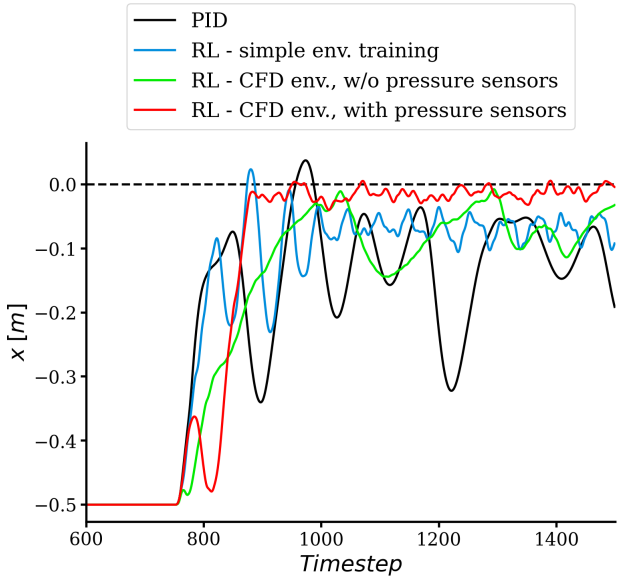


(b) Total reward during each episode

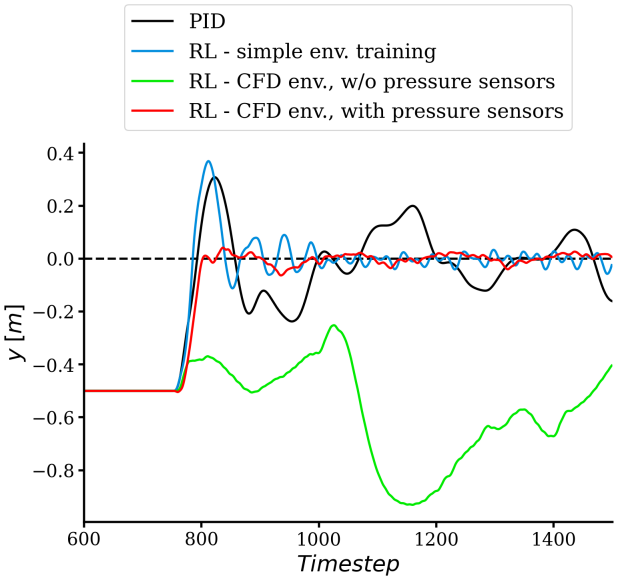


(c) Number of failed episodes in a moving window of 50 episodes

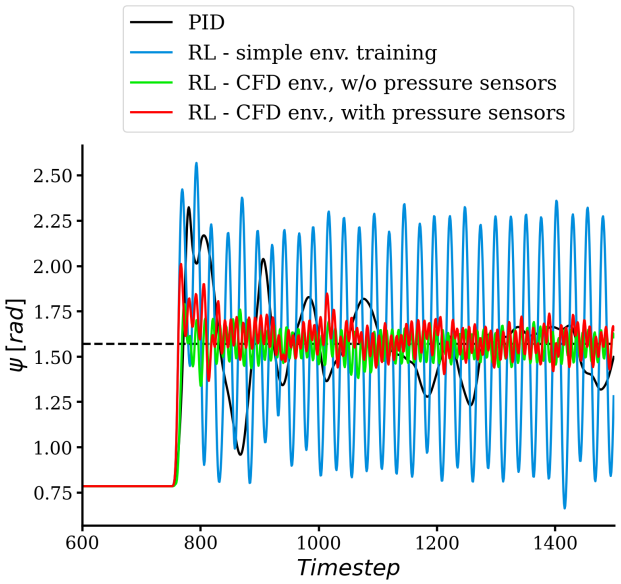
Figure 8: Time history of training the agents using inside CFD. Three instances of each variant (with or without pressure sensor feedback) have been trained to study the variability due to random seed of ANN weights and optimisation process. Agents used for final evaluation were selected based on the best moving average of the total reward and are highlighted with a black backdrop.



(a) X-position



(b) Y-position



(c) Heading

Figure 9: Time history of vehicle pose in the three considered degrees of freedom during the evaluation run. Initial part of the time trace corresponding to the CFD simulation start up not shown.

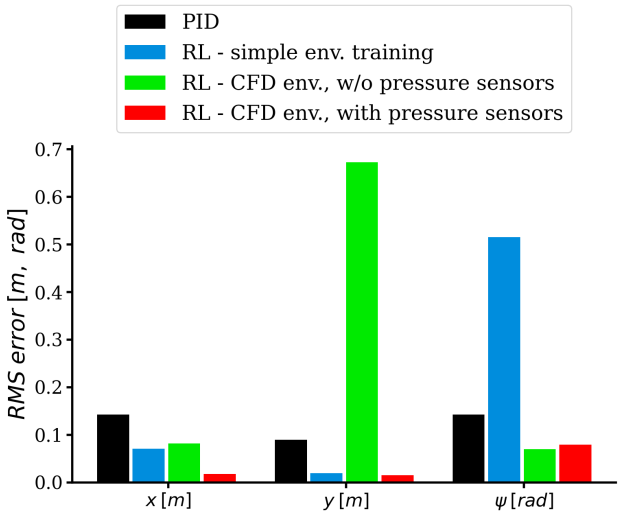


Figure 10: RMS errors in the three degrees of freedom attained during the evaluation run. These were computed during the final 500 time steps once all the agents have had the time to reach the objective.

4. CONCLUSIONS

Training of reinforcement learning agents inside CFD codes such as ReFRESCO is challenging in terms of the computational cost necessitated by the need to compute hundreds of thousands of time steps. Nevertheless, it has been successfully demonstrated to be possible and offer significant advantages by allowing the agent to learn inside a much more realistic environment than could be afforded by a simple manoeuvring model. Furthermore, having access to true flow field information allowed the agent to be augmented with a virtual set of pressure transducers that allowed it to estimate the forces acting on the vehicle. Doing so was shown to greatly improve the robustness and speed of the agent training process, as well as led to superior final performance when compared to an identical agent that did not posses the extra information. This shows a possibly significant advantage over classical optimal control strategies that typically rely on simplified models of system dynamics and cannot easily assimilate additional information of this sort. Further testing of the agent in previously unseen environments is needed in order to ascertain how robust and general the concept is, however.

Moreover, the current RL training strategy needs to be improved in order to make it lend itself better to application to expensive CFD computations. Possible steps include pre-training of the artificial neural networks using previously computed episodes, mirroring and rotating of the collected time steps in order to fill the replay buffer more quickly, and transfer learning by successively moving the agent from simpler and less expensive to more complex and computationally challenging environments. It would also be interesting to explore the possibility of hybrid RL-classical control whereby the RL agent would aim to correct the control outputs of the underlying linear controller. This might potentially allow the agent to deal well with rejecting disturbances and overcoming other nonlinear fluid dynamic effects without sacrificing the simplicity and sheer robustness of classical control strategies.

REFERENCES

Enrico Anderlini, Salman Husain, Gordon G. Parker, Mohammad Abusara, and Giles Thomas. Towards real-time reinforcement learning control of a wave energy converter. *Journal of Marine Science and Engineering*, 8(11):845, 2020. Publisher: MDPI.

Afsin Baran Bayezit. A generalized deep reinforcement learning based controller for heading keeping in waves. Master's thesis, Department of Ship Building and Ocean Engineering, Istanbul Technical University, Türkiye, 2022.

Simon Burmester, Guilherme Vaz, and Ould el Moctar. Towards credible CFD simulations for floating offshore wind turbines. *Ocean Engineering*, 209:107237, August 2020. ISSN 0029-8018. doi: 10.1016/j.oceaneng.2020.107237. URL https://www.sciencedirect.com/science/article/pii/S0029801820302870.

Eric A. D'Asaro and Geoffrey T. Dairiki. Turbulence Intensity Measurements in a Wind-Driven Mixed Layer. *Journal of Physical Oceanography*, 27(9):2009–2022, September 1997. ISSN 0022-3670, 1520-0485. doi: 10.1175/1520-0485(1997)027⟨2009:TIMIAW⟩2.0.CO;2. URL http://journals.ametsoc.org/doi/10.1175/1520-0485(1997)027<2009:TIMIAW>2.0.CO;2.

Dixia Fan, Liu Yang, Michael S. Triantafyllou, and George Em Karniadakis. Reinforcement learning for active flow control in experiments. *arXiv preprint arXiv:2003.03419*, 2020a.

Dixia Fan, Liu Yang, Zhicheng Wang, Michael S. Triantafyllou, and George Em Karniadakis. Reinforcement learning for bluff body active flow control in experiments and simulations. *Proceedings of the National Academy of Sciences*, 117(42):26091–26098, 2020b. Publisher: National Acad Sciences.

Thor I. Fossen. *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

Brian Gaudet, Richard Linares, and Roberto Furfaro. Deep reinforcement learning for six degree-of-freedom planetary landing. *Advances in Space Research*, 65(7):1723–1741, 2020. Publisher: Elsevier.

Peter Gunnarson, Ioannis Mandralis, Guido Novati, Petros Koumoutsakos, and John O. Dabiri. Learning efficient navigation in vortical flow fields. *Nature Communications*, 12(1):1–7, 2021. Publisher: Nature Publishing Group.

Kristian Gustavsson, Luca Biferale, Antonio Celani, and Simona Colabrese. Finding efficient swimming strategies in a three-dimensional chaotic flow by reinforcement learning. *The European Physical Journal E*, 40(12):1–6, 2017. Publisher: Springer.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, August 2018a. URL http://arxiv.org/abs/1801.01290. arXiv:1801.01290 [cs, stat].

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2018b. URL https://arxiv.org/abs/1812.05905.

Seungpyo Hong, Sejin Kim, and Donghyun You. Control of a fly-mimicking flyer in complex flow using deep reinforcement learning. *arXiv preprint arXiv:2111.03454*, 2021.

Omer Khalid, Guangbo Hao, Cian Desmond, Hamish Macdonald, Fiona Devoy McAuliffe, Gerard Dooly, and Weifei Hu. Applications of robotics in floating offshore wind farm operations and maintenance: Literature review and trends. *Wind Energy*, 25(11):1880–1899, 2022. ISSN 1099-1824. doi: 10.1002/we.2773. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/we.2773. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/we.2773.

Maarten Derk Klapwijk, Thomas P. Lloyd, and Guilherme Nuno Vasconcelos Beleza Vaz. The Development of a Partially Averaged Navier-Stokes KSKL Model. *Journal of Fluids Engineering*, September 2021. ISSN 0098-2202. doi: 10.1115/1.4052484. URL https://doi.org/10.1115/1.4052484. _eprint: https://asmedigitalcollection.asme.org/fluidsengineering/article-pdf/doi/10.1115/1.4052484/6759213/fe-21-1272.pdf.

Andrea Lario, Romit Maulik, Oliver T. Schmidt, Gianluigi Rozza, and Gianmarco Mengaldo. Neural-network learning of SPOD latent dynamics. *Journal of Computational Physics*, 468:111475, November 2022. ISSN 00219991. doi: 10.1016/j.jcp.2022.111475. URL http://arxiv.org/abs/2110.09218. arXiv:2110.09218 [cs, math].

Myoung Hoon Lee and Jun Moon. Deep reinforcement learning-based model-free path planning and collision avoidance for uavs: A soft actor–critic with hindsight experience replay approach. *ICT Express*, 2022. ISSN 2405-9595. doi: https://doi.org/10.1016/j.icte.2022.06.004. URL https://www.sciencedirect.com/science/article/pii/S2405959522000935.

Shaopeng Li, Reda Snaiki, and Teng Wu. Active simulation of transient wind field in a multiple-fan wind tunnel via deep reinforcement learning. *Journal of Engineering Mechanics*, 147(9):04021056, 2021. Publisher: American Society of Civil Engineers.

Artur K. Lidtke, Maarten Klapwijk, and Thomas Lloyd. Scale-Resolving Simulations of a Circular Cylinder Subjected to Low Mach Number Turbulent Inflow. *Journal of Marine Science and Engineering*, 9(11), 2021. ISSN 2077-1312. URL https://www.mdpi.com/2077-1312/9/11/1274.

Andreas B. Martinsen, Anastasios M. Lekkas, Sébastien Gros, Jon Arne Glomsrud, and Tom Arne Pedersen. Reinforcement learning-based tracking control of usvs in varying operational conditions. *Frontiers in Robotics and AI*, 7:32, 2020. Publisher: Frontiers Media SA.

Andrea Munafò, Jan Sliwka, Gabriele Ferri, Arjan Vermeij, Ryan Goldhahn, Kevin LePage, Joao Alves, and John Potter. Enhancing AUV localization using underwater acoustic sensor networks: Results in long baseline navigation from the COLLAB13 sea trial. In *2014 Oceans-St. John's*, pages 1–7. IEEE, 2014.

Guido Novati, Lakshminarayanan Mahadevan, and Petros Koumoutsakos. Controlled gliding and perching through deep-reinforcement-learning. *Physical Review Fluids*, 4(9):093902, 2019. Publisher: APS.

Kay I. Ohshima. Near-surface circulation and tidal currents of the Okhotsk Sea observed with satellite-tracked drifters. *Journal of Geophysical Research*, 107(C11):3195, 2002. ISSN 0148-0227. doi: 10.1029/2001JC001005. URL http://doi.wiley.com/10.1029/2001JC001005.

M. Orlic, M. Gacic, and P. E. Laviolette. The currents and circulation of the Adriatic Sea. *Oceanologica acta*, 15(2):109–124, 1992. Publisher: Gauthier-Villars.

Chen-Huan Pi, Yi-Wei Dai, Kai-Chun Hu, and Stone Cheng. General Purpose Low-Level Reinforcement Learning Control for Multi-Axis Rotor Aerial Vehicles. *Sensors*, 21(13):4560, 2021. Publisher: MDPI.

Jean Rabault, Feng Ren, Wei Zhang, Hui Tang, and Hui Xu. Deep reinforcement learning in fluid mechanics: A promising method for both active flow control and shape optimization. *Journal of Hydrodynamics*, 32(2):234–246, 2020. Publisher: Springer.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `http://jmlr.org/papers/v22/20-1364.html`.

Feng Ren, Jean Rabault, and Hui Tang. Applying deep reinforcement learning to active flow control in weakly turbulent conditions. *Physics of Fluids*, 33(3):037121, 2021. Publisher: AIP Publishing LLC.

Emanuel P. P. Soares-Ramos, Lais de Oliveira-Assis, Raúl Sarrias-Mena, and Luis M. Fernández-Ramírez. Current status and future trends of offshore wind power in Europe. *Energy*, 202: 117787, July 2020. ISSN 0360-5442. doi: 10.1016/j.energy.2020.117787. URL `https://www.sciencedirect.com/science/article/pii/S036054422030894X`.

Kantapon Tanakitkorn, Philip A. Wilson, Stephen R. Turnock, and Alexander B. Phillips. Depth control for an over-actuated, hover-capable autonomous underwater vehicle with experimental verification. *Mechatronics*, 41:67–81, 2017. Publisher: Elsevier.

Kantapon Tanakitkorn, Philip A. Wilson, Stephen R. Turnock, and Alexander B. Phillips. Sliding mode heading control of an overactuated, hover-capable autonomous underwater vehicle with experimental verification. *Journal of Field Robotics*, 35(3):396–415, 2018. Publisher: Wiley Online Library.

Hongwei Tang, Jean Rabault, Alexander Kuhnle, Yan Wang, and Tongguang Wang. Robust active flow control over a range of Reynolds numbers using an artificial neural network trained through deep reinforcement learning. *Physics of Fluids*, 32(5):053605, 2020. Publisher: AIP Publishing LLC.

Yodyium Tipsuwan and Phakhachon Hoonsuwan. Design and implementation of an AUV for petroleum pipeline inspection. In *2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 382–387, October 2015. doi: 10.1109/ICITEED.2015.7408976.

Siddhartha Verma, Guido Novati, and Petros Koumoutsakos. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proceedings of the National Academy of Sciences*, 115(23):5849–5854, 2018. Publisher: National Acad Sciences.

Jonathan Viquerat, Philippe Meliga, and Elie Hachem. A review on deep reinforcement learning for fluid mechanics: an update. *arXiv preprint arXiv:2107.12206*, 2021.

Marco Vona and Eric Lauga. Stabilizing viscous extensional flows using reinforcement learning. *Physical Review E*, 104(5):055108, 2021. Publisher: APS.

Joohyun Woo, Chanwoo Yu, and Nakwan Kim. Deep reinforcement learning-based controller for path following of an unmanned surface vehicle. *Ocean Engineering*, 183:155–166, July 2019. ISSN 00298018. doi: 10.1016/j.oceaneng.2019.04.099. URL `https://linkinghub.elsevier.com/retrieve/pii/S0029801819302203`.

Chu-Jou Wu. *6-dof modelling and control of a remotely operated vehicle*. PhD Thesis, Flinders University, College of Science and Engineering., 2018.

Jian Xu, Fei Huang, Di Wu, Yunfei Cui, Zheping Yan, and Xue Du. A learning method for auv collision avoidance through deep reinforcement learning. *Ocean Engineering*, 260:112038, 2022. ISSN 0029-8018. doi: https://doi.org/10.1016/j.oceaneng.2022.112038. URL `https://www.sciencedirect.com/science/article/pii/S0029801822013683`.

Yuemin Zheng, Jin Tao, Qinglin Sun, Hao Sun, Zengqiang Chen, Mingwei Sun, and Guangming Xie. Soft actor–critic based active disturbance rejection path following control for unmanned surface vessel under wind and wave disturbances. *Ocean Engineering*, 247:110631, 2022. ISSN 0029-8018. doi: https://doi.org/10.1016/j.oceaneng.2022.110631. URL `https://www.sciencedirect.com/science/article/pii/S0029801822000968`.

Mingxi Zhou, Ralf Bachmayer, and Brad De Young. Underwater acoustic-based navigation towards multi-vehicle operation and adaptive oceanographic sampling. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6091–6097. IEEE, 2017.