Journal of Marine Science and Application (2026) 25: 728-744
https://doi.org/10.1007/s11804-026-00820-x

RESEARCH ARTICLE

Transfer Learning for Deep Reinforcement Learning-Based Path Following of Autonomous Surface Vessels

Aniket Malviya[1], Suresh Rajendran[1] and Xueqian Zhou[2]

Received: 24 May 2025 / Accepted: 11 March 2026

Abstract

Deep Reinforcement Learning (DRL) offers a powerful, model-free, and data-driven approach for the navigation and control of Autonomous Surface Vessels (ASVs). The primary challenge, however, lies in the extensive training required for an agent to converge to an effective policy within a complex simulation, leading to significant computational overhead. This paper presents a multi-stage training framework that uses Transfer Learning to pass knowledge between different simulation models, resulting in a highly robust DRL controller for ASVs. The proposed framework utilizes the Deep Deterministic Policy Gradient (DDPG) algorithm to develop the data-driven controller. First, a foundational policy is efficiently learned using a simplified first-order Nomoto dynamics and second-order Nomoto dynamics, which captures the fundamental vessel dynamics. This pre-trained policy is then transferred to a complex, nonlinear Manoeuvring Modelling Group (MMG) model, significantly accelerating training convergence. Subsequently, the agent is fine-tuned within the MMG simulation with environmental disturbances. The models are evaluated on various trajectories during testing to ensure robust performance. The accuracy of the DRL controller is assessed by measuring heading error (e_ψ) and cross-track error (y_e). A traditional Proportional-Integral-Derivative (PID) controller is implemented and compared to benchmark the DRL controller's effectiveness, to highlight the relative advantages and limitations of each approach.

Keywords Deep reinforcement learning (DRL); Autonomous surface vessels (ASVs); Deep deterministic policy gradient (DDPG); Transfer learning; Proportional-integral-derivative (PID) controller; Line of sight (LOS) guidance algorithm

1 Introduction

Autonomous Surface Vessels (ASVs) represent a promising frontier in maritime operations, offering the potential for efficient and autonomous navigation across vast expanses of water. Developing robust control schemes for ASVs is crucial to ensuring their safe and effective deployment in real-world scenarios. Traditional model-based control strategies-such as PID, LQR, and Model Predictive Control-have long been used for ship navigation tasks like heading control and path following (Katebi and Moradi, 2001; Tomera, 2017; Perera et al., 2014).

Due to its model-free and data-driven capabilities, deep reinforcement learning (DRL) has emerged as a promising approach for developing control systems. DRL-based controllers learn optimal policies through environmental interaction, making them suitable for complex tasks (Puterman, 2014; Busoniu et al., 2018). RL-based methods have demonstrated promise in a wide range of maritime tasks, including path planning, collision avoidance, and path following. In particular, Deep Reinforcement Learning (DRL) algorithms, such as Deep Deterministic Policy Gradient (DDPG), are well-suited for continuous control problems like determining rudder angle rates in marine vessels. However, despite their adaptability, DRL approaches face a critical challenge: the slow and unstable learning phase in the initial training stage, which can lead to unsafe actions if implemented directly on real ships.

In this paper, we investigate the effectiveness of transfer learning in accelerating DRL training for the path-following control of a KVLCC2 tanker model under calm-water

Article Highlights

- Demonstrates a two-stage transfer learning framework for DRL-based path following of autonomous surface vessels.
- Achieves up to 5× faster training convergence compared to training from scratch.
- Shows that pre-training with Second-Order Nomoto models mitigates negative transfer and improves learning stability.
- Benchmarks DRL controllers against a tuned PID controller under calm and wave-disturbance conditions, highlighting trade-offs in accuracy and robustness.

✉ Suresh Rajendran
sureshr@iitm.ac.in

[1] Ocean Engineering, Indian Institute of Technology Madras, Chennai 600036, India

[2] Harbin Engineering University, Shipbuilding Engineering University, Harbin 150001, China

and wave-disturbance conditions. The proposed approach trains a DDPG-based controller in three configurations: without transfer learning, with transfer from a First Order Nomoto (FON) model, and with transfer from a Second Order Nomoto (SON) model. The goal is to evaluate whether knowledge transfer from these simplified models can significantly reduce training time while improving convergence stability in the complex MMG environment. Our findings reveal that this approach accelerates convergence by up to five times compared to training from scratch, demonstrating a substantial improvement in training efficiency and ensuring the safety of the ship.

The main contributions of this work are:

• A transfer learning framework for DRL-based ASV path following, comparing the benefits of transferring from first- and second-order Nomoto models.

• A comprehensive evaluation under calm-water and wave-disturbance conditions, assessing heading error, cross-track error, and convergence time.

• A benchmark comparison against a tuned Proportional-Integral-Derivative (PID) controller to provide a baseline for the DRL agents' performance.

The remainder of the paper is organized as follows: Section 2 describes the ship models. Section 3 presents the line-of-sight guidance algorithm. Section 4 outlines the conceptual background of reinforcement learning, transfer learning and PID control. Section 5 presents the experimental setup and methodology. Section 6 discusses the training and testing results, and Section 7 concludes the paper with key findings and directions for future work.

2 Ship dynamics

This section delves into the ship dynamics models for training our deep reinforcement learning controller. We utilize three distinct dynamics: the First Order Nomoto, Second Order Nomoto and the MMG KVLCC2 tanker ship dynamics. Both models are derived from the MMG (Maneuvering Modeling Group) model (Yasukawa and Yoshimura, 2015; Paramesh and Rajendran, 2021). The derivation of these models involves using regression methods to accurately capture the ship's dynamic behaviour. For consistency across our simulations, we standardized key parameters: the model ship has a length of 2.9 meters, a velocity of 0.78 meters per second, and the rudder angle varies between -35 and 35 degrees

2.1 Maneuvering modeling group (MMG) dynamics

The MMG (Maneuvering Modeling Group) model utilized in this project, sourced from the research paper by Yasukawa and Yoshimura (2015) and further adapted by Paramesh and Rajendran (2021) and Sandeepkumar et al., (2021) serves as a robust mathematical framework for predicting ship manoeuvring motions. The numerical simulations are carried out for an L3 model of a KVLCC2 tanker (model scale 1 : 110). The main particulars of the ship are given in Table 1.

Table 1 Main particulars of the ship and the model

Particular	Full scale	L3-model
Length between perpendiculars (m)	320	2.909
Breadth (m)	58	0.527
Draft (m)	20.8	0.189
Displacement (m^3)	312600	0.235
Longitudinal center of gravity (m)	11.2	0.102
Block coefficient	0.810	0.810
Rudder height (m)	15.80	0.144
Rudder area (m)	112.5	0.00928
Propeller diameter (m)	9.86	0.090

The MMG model is specifically applied to the KVLCC2 tanker ship and considers three degrees of freedom (3-DoF): surge (u), sway (v), and yaw (ψ). The equations of motion for the surge-sway-yaw system can be expressed as:

$$\begin{cases} \left(m + m_x\right)\dot{u} - \left(m + m_y\right)vr - x_G m r^2 = X_H + X_R + X_P + X_W \\ \left(m + m_y\right)\dot{v} + \left(m + m_x\right)ur + x_G m\dot{r} = Y_H + Y_R + Y_W \\ \left(I_{z_G} + x_G^2 m + J_z\right)\dot{r} + x_G m\left(\dot{v} + ur\right) = N_H + N_R + N_W \end{cases} \tag{1}$$

where the right-hand side terms represent contributions from:

- H: Hull forces and moments,
- R: Rudder forces and moments,
- P: Propeller forces and moments,
- W: Wave drift forces and moments.
- X: Surge Force
- Y: Sway Force
- N: Yaw Moment

where m is the ship's mass, m_x and m_y are the added masses in surge and sway, I_{z_G} is the moment of inertia, J_z is the added moment of inertia, and x_G is the longitudinal center of gravity.

To ensure this manuscript remains self-contained, the fundamental assumptions and the complete resolving formulae for the right-hand side hydrodynamic forces of the MMG model are detailed in Appendix A.

The kinematic equations of motions are:

$$\begin{cases} \dot{X} = u\cos\psi - v\sin\psi \\ \dot{Y} = u\sin\psi + v\cos\psi \\ \dot{\psi} = r \end{cases} \tag{2}$$

2.2 System identification and data collection

To derive the simplified First-Order and Second-Order Nomoto models essential for the pre-training stage, a system identification procedure is performed using data generated from the comprehensive nonlinear MMG model. To accurately capture the transient steering dynamics required for course-keeping and path-following tasks, a standard 10°/10° zig-zag maneuver is simulated. During the simulated 10°/10° zig-zag maneuver, the time-series data for the rudder angle (δ), yaw rate (r), and yaw acceleration ($\dot{r}$) are recorded at a fixed sampling interval. To estimate the unknown coefficients of the Nomoto models, the system identification task is structured as an overdetermined linear regression problem.

By arranging the recorded time-series data into a matrix format, the dynamic equations can be expressed generically as:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{B} \tag{3}$$

where $\boldsymbol{A}$ is the observation matrix containing the recorded state variables (such as yaw rate and rudder angle at each time step), $\boldsymbol{B}$ is the response vector (such as the corresponding yaw acceleration), and $\boldsymbol{x}$ is the vector of unknown Nomoto parameters to be identified.

Because the observation matrix $\boldsymbol{A}$ is non-square due to the large number of sampled data points, the system cannot be solved by simple inversion. Instead, the optimal parameter vector $\boldsymbol{x}$ that minimizes the sum of squared residuals is computed using the Moore-Penrose pseudo-inverse:

$$\boldsymbol{x} = \left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B} \tag{4}$$

This least-squares approach provides a computationally efficient and highly robust method for extracting the First-Order and Second-Order Nomoto parameters directly from the simulated zig-zag data. The accuracy of this identification process is visually validated in Figure 1, which demonstrates the high-fidelity fit of the resulting linear models against the nonlinear MMG reference data. These resulting linear models subsequently serve as the foundational training environments for the DRL controller.

2.3 First order nomoto dynamics

The First-Order Nomoto Model adopted in this project provides a simplified linear representation of the ship's steering dynamics. This model does not dynamically account for changes in surge velocity; it only maps the relationship between the rudder angle and the resulting yaw rate.

The governing equation for the First-Order Nomoto model is defined as:

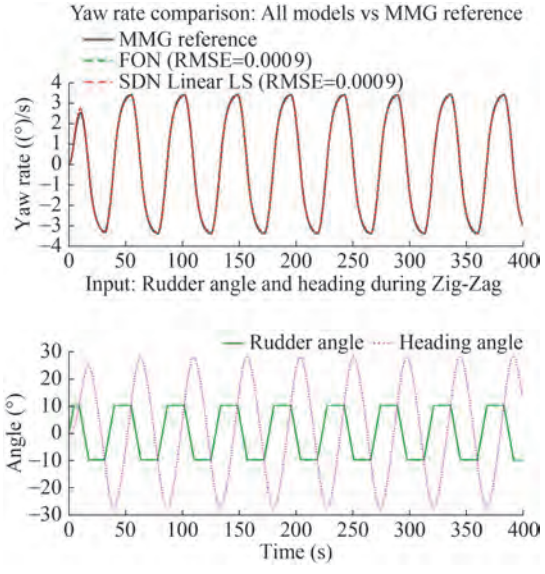


Figure 1 System identification using zig-zag maneuver

$$T\dot{r} + r = k\delta \tag{5}$$

To derive the parameters (T, k), the time-series data $(r, \dot{r}, \delta)$ obtained from the nonlinear MMG model during the 10°/10° zig-zag maneuver is utilized. As described in Section 2.2, Equation (5) is discretized and formulated into an overdetermined linear regression problem ($\boldsymbol{A}\boldsymbol{x} = \boldsymbol{B}$). Solving this system yields the identified coefficients, providing a time constant (T) of 5.005 5 s and a rudder gain (k) of 0.358 0/s.

Unlike the MMG model, the First Order Nomoto Model does not account for the reduction in velocity during the turning circle simulation. Therefore, the reduced velocity observed in the MMG model simulation is used in subsequent simulations with the Nomoto model to ensure a fair comparison.

2.4 Second order nomoto dynamics

The Second-Order Nomoto Model, similar to the First-Order model, is derived from the comprehensive MMG dynamics to serve as a high-quality but computationally simplified control environment. While this model also operates under a constant surge velocity, it provides a much more detailed representation of the ship's transient steering dynamics—such as phase lag and overshoot—by incorporating yaw acceleration ($\ddot{r}$) and the rate of rudder execution ($\dot{\delta}$). The governing equation for the Second-Order Nomoto model is defined as:

$$T_1T_2\ddot{r} + \left(T_1 + T_2\right)\dot{r} + r = k\left(\delta + T_3\dot{\delta}\right) \tag{6}$$

To identify the physical parameters (T_1, T_2, T_3, k), the regression formulation established in Section 2.2 is

expanded to account for the higher-order derivatives extracted from the 10°/10° zig-zag maneuver data. Equation (6) is rearranged into the intermediate form $\alpha\ddot{r} + \beta\dot{r} + r = k\delta + \gamma\dot{\delta}$, where the grouped coefficients correspond to $\alpha = T_1T_2$, $\beta = T_1 + T_2$ and $\gamma = kT_3$. The observation matrix is augmented accordingly, and an initial coefficient vector is solved via the pseudo-inverse method. The preliminary physical parameters are then extracted from the roots of the characteristic equation formed by α and β. The final optimized parameters obtained are $T_1 = 4.704\,3$ s, $T_2 = 0.963\,7$ s, $T_3 = 0.757\,2$ s, and $k = 0.352\,3$/s. As required for dynamic consistency, the resultant sum $T_1 + T_2 - T_3 = 4.910\,8$ s closely matches the First-Order time constant ($T = 5.005\,5$ s).

To illustrate the performance and inherent limitations of these simplified dynamics, Figure 2 compares the models during a 10° turning circle maneuver. While both the FON and SON models successfully capture the steady-state yaw rate and heading angles of the MMG reference, they fail to reproduce the significant surge velocity drop caused by drag and sway-yaw coupling. Consequently, the spatial trajectory deviates over time. This limitation explicitly highlights why the linear Nomoto models are used solely for pre-training fundamental steering policies (Stage 1), necessitating a transfer to the fully coupled, nonlinear MMG environment (Stage 2) to achieve robust, real-world path-following control.

3 Line of sight (LoS) algorithm

The Line of Sight (LoS) algorithm, developed following the methodologies of Lekkas and Fossen, 2013 calculates the Heading Error (e_ψ) and Cross-Track Error (y_e) based on the vessel's current state and its distance from the desired trajectory. Cross Track Error and Heading Error is calculated as follows:

$$\begin{aligned} y_e &= -(x - x_k)\sin\gamma_p + (y - y_k)\cos\gamma_p \\ \psi_d &= \gamma_p + \arctan\left(-\frac{y_e}{\Delta}\right) \\ e_\psi &= \psi_d - \psi \end{aligned} \tag{7}$$

where,

$$\gamma_p = \text{atan2}(y_{k+1} - y_k, x_{k+1} - x_k)$$

and, (x, y) is the ship's current position, (x_k, y_k) are the coordinates of the current waypoint. Look-ahead distance (Δ) is set to two times the ship's length.

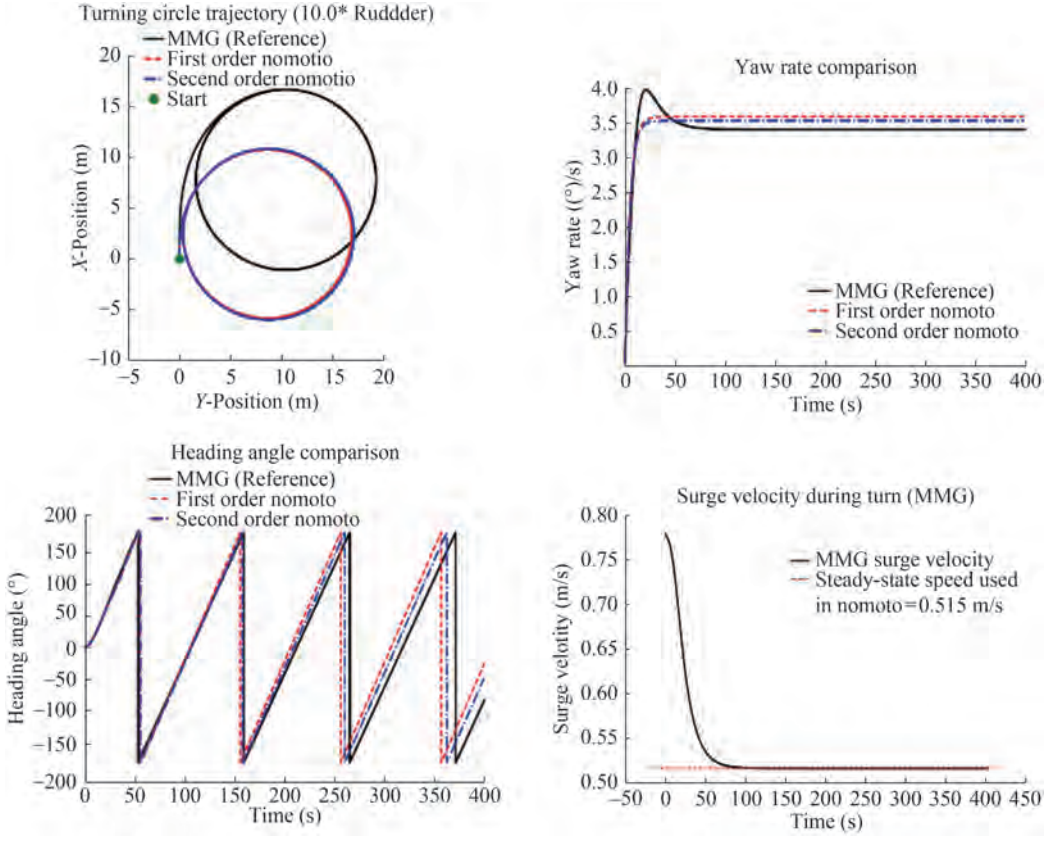


Figure 2 Comparison of the MMG, FON, and SON models during a 10° turning circle maneuver

4 Conceptual background

4.1 Deep reinforcement learning

Reinforcement Learning (RL) is a type of machine learning where agents learn to make decisions by performing actions in an environment to maximize cumulative rewards (Sutton and Barto, 2018). The agent's goal is to learn a policy that maximizes the cumulative reward over time by performing actions and receiving feedback in the form of numerical rewards. Deep Reinforcement Learning (DRL) extends this paradigm by using deep neural networks (DNNs) to approximate the complex policy and value functions that map an agent's state to its optimal actions. This approach makes DRL exceptionally effective for solving high-dimensional and continuous control problems.

In the context of a control system (Kiumarsi et al., 2017), the DRL framework is structured as follows: Reinforcement Learning (RL) -based control for a ship involves leveraging machine learning techniques to enable the vessel to learn optimal actions in a dynamic environment. In this approach, acting as an agent, the ship interacts with its surroundings and receives feedback through rewards or penalties based on its actions. Through iterative learning processes, the RL algorithm adapts and refines the ship's control policies to maximize cumulative rewards, allowing it to navigate and make decisions in complex maritime scenarios autonomously.

In this study, we employ the Deep Deterministic Policy Gradient (DDPG) algorithm, which is an actor-critic method capable of operating in continuous action spaces—ideal for rudder control in ASVs. The choice of DDPG is supported by prior comparative analysis in (Sivaraj et al., 2023), where DDPG demonstrated the lowest average cross-track error (y_e) among tested controllers, outperforming PPO, TD3, and SAC. Although DDPG exhibited a slightly higher standard deviation in y_e compared to PPO—indicating marginally larger overshoots—it consistently maintained superior path-following accuracy. The state space vector provided to the neural network includes r, e_ψ, and y_e. The agent's output is the rudder rate, which represents a continuous action space. A feed-forward neural network (FFNN) with two hidden layers of 128 neurons each, and bias terms, is used for both actor and critic networks, and it utilizes the tanh activation function in the output layer to map its output to the symmetric limits of the rudder rate (e.g., [−15°/s, 15°/s]). According to Froude scaling laws, angular rates scale with $\sqrt{\lambda}$, where $\lambda = 110$. Therefore, a model-scale rate of 15°/s corresponds to approximately 1.43°/s at full scale. While classification societies generally mandate a minimum physical capability of 2.23°/s, this restricted rate is implemented deliberately as a conservative operational bound for the DRL agent, preventing it from learning overly aggressive steering policies and ensuring smooth hardware execution.

4.1.1 Deep deterministic policy gradient (DDPG)

The Deep Deterministic Policy Gradient (DDPG) algorithm is a model-free, off-policy reinforcement learning method designed for continuous action spaces (Lillicrap et al., 2015; Silver et al., 2014). It merges concepts from Deep Q-Networks (DQN) (Mnih et al., 2013; Mnih et al., 2015) and policy gradient methods, adopting an actor-critic architecture to simultaneously learn a policy and a value function.

The actor network $\mu(s|\theta^\mu)$ maps states to deterministic actions, while the critic network $Q(s, a|\theta^Q)$ evaluates the quality of state-action pairs. To stabilize training, DDPG maintains two additional target networks—a target actor μ' and target critic Q'-which are time-delayed copies of the main networks, updated via soft updates to avoid rapid oscillations.

As an off-policy algorithm, DDPG utilizes a replay buffer to store a large history of transitions (state, action, reward, next state). During training, mini-batches are sampled from this buffer, allowing the algorithm to learn from a diverse set of uncorrelated experiences, which significantly improves learning stability. At each time-step, the agent observes the current state s_t, selects an action $a_t = \mu(s_t) + N_t$ where N_t represents exploration noise (commonly Ornstein-Uhlenbeck noise is used for temporally correlated exploration), and receives a reward r_t and the next state s_{t+1}. Each transition (s_t, a_t, r_t, s_{t+1}) is stored in a replay buffer, from which mini-batches are sampled to break temporal correlations in the training data.

The critic network is trained to minimize the mean squared error between its Q-value prediction and the target Q-value computed from the Bellman equation:

$$y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1}|\theta^{\mu'}\right)|\theta^{Q'}\right) \tag{8}$$

The actor network is updated using the deterministic policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q\left(s, a|\theta^Q\right)\Big|_{a=\mu(s)} \nabla_{\theta^\mu}\mu\left(s|\theta^\mu\right) \tag{9}$$

Finally, the target networks are updated with soft updates:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta' \tag{10}$$

where $\tau \ll 1$ ensures slow tracking of the main networks.

This combination of a replay buffer, target networks, and soft updates provides the necessary stability for DDPG to effectively learn complex control policies in continuous domains, making it well-suited for tasks such as rudder control in ASVs, where smooth and precise action outputs are essential.

4.2 Transfer learning

Transfer Learning (TL) is a machine learning paradigm that addresses this challenge by leveraging knowledge gained from one problem (a source task) to improve performance on a different but related problem (a target task) (Pan and Yang, 2009). Instead of training a model from scratch for every new problem, TL reuses the learned representations—such as features, parameters, or policies—from a pre-trained model and adapts them to the new context. This approach is particularly valuable when the target task has limited training data, is computationally expensive to simulate, or requires faster convergence to a high-performing policy.

In the domain of deep reinforcement learning, transfer learning is a powerful strategy for accelerating agent training. This is achieved by transferring knowledge such as policy weights, value function parameters, or learned feature representations from a simpler or related environment to a more complex target environment (Taylor and Stone, 2009). By doing so, the agent can start with a near-optimal initial policy instead of a random one, significantly shortening the exploration phase and improving learning stability.

For Autonomous Surface Vehicles (ASVs), TL is highly beneficial because real-world maritime environments are dynamic, stochastic, and expensive to replicate in simulation. Training directly in a high-fidelity ship simulator that includes nonlinear hydrodynamics, wave disturbances, and environmental noise can be computationally intensive and slow. Instead, an RL agent can be initially trained in a simplified, computationally efficient model—such as a First-Order Nomoto model in calm-water conditions—to learn basic path-following and control dynamics. The learned policy can then be transferred to a more complex, high-fidelity simulation (e.g., MMG model with environmental disturbances) for fine-tuning.

4.3 Proportional-integral-derivative (PID) controller

The Proportional-Integral-Derivative (PID) controller is one of the most widely used control strategies in engineering due to its simplicity, intuitive design, and effectiveness in a broad range of applications. Its primary function is to minimize the error between a measured process variable and a desired set-point by adjusting a control output. In the context of ship control for path following, the PID controller uses the heading error ($e_\psi(t)$) as input and generates the rudder angle ($\delta(t)$) as output, which is then applied to the ship's dynamics The PID control law is expressed as:

$$\delta(t) = K_p e_\psi(t) + K_i \int_0^t e_\psi(\tau)\,\mathrm{d}\tau + K_d \frac{\mathrm{d}e_\psi(t)}{\mathrm{d}t} \quad (11)$$

where K_p, K_i, and K_d represent the proportional, integral, and derivative gains respectively.

4.4 Second order wave drift forces

A 2D potential flow method based on strip theory is used to estimate the steady drift forces. Thus, the methodology adopts the classical potential-flow assumptions in combination with the strip-theory approximation. The body potential is assumed to be much smaller than the incident wave potential based on the weak scatter approach. The method computes second-order mean drift forces using first-order linear potentials that satisfy the linear free-surface condition. The second-order effects arise through time-averaging the quadratic product of linear first-order solution. The total first-order potential is composed of incident, diffraction, and radiation components. The steady second-order force is obtained by evaluating the surface integral of the quadratic product of the first-order potential and its normal derivative. The evaluation of this surface integral is simplified by adopting the strip-theory approximation, which converts the 3D hydrodynamic quantities into 2D sectional integrals. The final drift force separates into a Havelock (motion × Froude-Krylov forcing) term, a diffraction-radiation interaction term, and a purely sectional term. The complete derivation can be found in Salvesen (1974) and briefly summarized in appendix B. Paramesh and Rajendran (2021) and Sandeepkumar et al. (2021) calculated the 2nd order wave drift forces and moments acting on a KVLCC2 tanker and validated against the experimental results available from the literature. Initially, the wave drift forces are calculated for a range of frequencies and headings and stored in the database. During the manoeuvring simulation, the wave forces are interpolated for the exact ship heading and frequency and added to the 3 DoF equation of motion. A similar approach is followed in this paper.

5 Methodology

In this paper, a two-stage transfer learning framework is proposed to optimize training efficiency and final policy performance, thereby developing a robust DRL-based controller for ASV path-following. The main idea is to split the training process into two steps so that the agent can learn basic control skills in a low-complexity, computationally efficient environment before adjusting to the uncertainties and dynamics of a high-fidelity ship model. The following sections describe this framework's entire architecture, from initial training to final evaluation.

5.1 Reinforcement learning setup

The implementation of the DRL framework is carried out using the Python programming language; the core of

the training process is built upon the OpenAI Gym toolkit, which provides a standardized interface for reinforcement learning environments, and the Stable Baselines3 library, a set of reliable implementations of state-of-the-art RL algorithms.

The Deep Deterministic Policy Gradient (DDPG) algorithm was used for this study because it works well for problems that require continuous control, like controlling the rudder on Autonomous Surface Vessels (ASVs). DDPG uses an actor-critic architecture, where the actor network provides continuous commands for the rudder rate and the critic network validates the quality of state-action pairs. The actor and critic are both feed-forward neural networks with two hidden layers, each with 128 neurons, and bias terms in all layers. The tanh activation function is used in the actor network's output layer to smoothly map the control output within the symmetric rudder rate limits of [−15°/s, 15°].

All training experiments were conducted on a workstation equipped with an Nvidia GeForce GTX 1650 GPU (4 GB VRAM) and an AMD Ryzen 5 4600H CPU, which provided the necessary computational power for the numerous simulation episodes.

5.1.1 Environment

To facilitate the agent's training, a custom path-following environment is developed in Python, adhering to the OpenAI Gym interface. For each new training episode, a unique path is randomly generated. As shown in Figure 3, the path consists of ten waypoints arranged at random angles to create diverse and challenging navigation scenarios. The distance between the waypoints is roughly fifteen times the length of the ship. Both the ship's starting position and its initial yaw angle (direction of velocity) are chosen randomly to further diversify the training conditions.

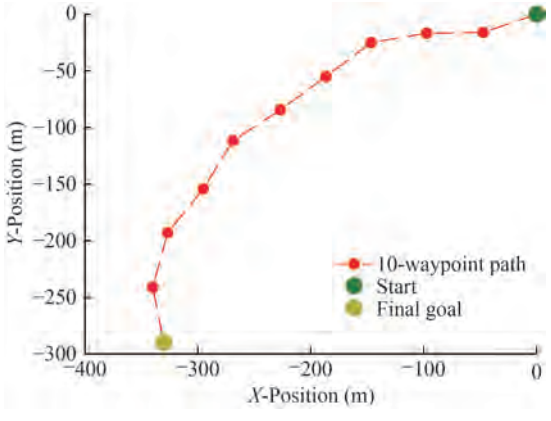


Figure 3 Training path for a random episode

An episode is concluded when the agent successfully reaches the final waypoint or when one of the following termination criteria is met:

• Exceeding maximum allowable steps:

$$\text{steps} = 1.2 \times \frac{D}{u \cdot \Delta t}$$

where, D is pathlength, u is ship's velocity, and Δt is time step.

• Excessive heading deviation:

$$e_\psi > 175°$$

• Excessive Cross-Track Error:

$$y_e > 30\ \text{m}$$

5.1.2 Reward function

The reward function employed is linear, incorporating yaw rate, Heading Error (e_ψ), and Cross Track Error (y_e) as variables. The objective is to minimize HE and CTE to encourage the agent to adhere closely to the path and maximize the reward. Additionally, including the yaw rate in the reward function minimizes sharp turns, promoting a more consistent and stable navigation strategy. The reward function incorporates heading error (e_ψ), cross-track error (y_e), and yaw rate (r) as variables:

$$R = 1 - a\left|e_\psi\right| - b\left|y_e\right| - c\left|r\right| \tag{12}$$

where, a, b, and c are weighting coefficients.

The main goal is to reduce both e_ψ and y_e, which will motivate the agent to adhere closely to the intended course. The inclusion of the yaw rate term penalizes excessive rotational motion, discouraging sharp turns and promoting a smoother, more stable navigation strategy.

Before being used in the reward computation, e_ψ is normalized from its raw range [−180°/s, +180°/s] and clipped to $[-1, 1]$. Similarly, y_e is clipped to $[-1, 1]$ to prevent extreme deviations from disproportionately affecting the reward signal. This normalization ensures balanced contributions of all terms in the reward.

5.2 Source policy learning in a calm environment

Stage 1 of the proposed framework focuses on enabling the agent to acquire fundamental path-following skills in a simplified, disturbance-free environment. This serves as the source task for transfer learning. The environment is modeled using the First-Order Nomoto ship dynamics (Equation (3) in Section 2.2) and Second-Order Nomoto dynamics (Equation (4) in Section 2.2). The training is conducted in calm water, with no environmental disturbances such as wind, waves, or currents. The state vector provided to the agent consists of:

• Yaw rate (r): rate of change of vessel heading.

• Heading Error (e_ψ): angular difference between the desired and actual heading.

• Cross-Track Error (y_e): lateral deviation from the desired path.

The action is the rudder rate, defined in continuous space and constrained to [−15°/s, +15°/s]. This continuous action space is particularly well-suited for the Deep Deterministic Policy Gradient (DDPG) algorithm employed in this work.

Through iterative interaction with this simplified environment, the agent learns a baseline policy that serves as a basis for transfer to a more complex environment in Stage 2.

5.3 Robustness fine-tuning via knowledge transfer

Stage 2 of the proposed framework focuses on adapting the baseline policy learned in the simplified environment to a complex ship-dynamics model that better reflects real-world operating conditions. The Maneuvering Modeling Group (MMG) model, described in Section 2.1, is employed in this stage. The DDPG agent's actor and critic networks are initialized in the MMG environment using the pre-trained weights from the calm-water expert policy acquired in Stage 1 to facilitate knowledge transfer.

To evaluate the effectiveness of transfer under different disturbance conditions, two separate fine-tuning experiments are conducted:

• Fine-Tuning in Calm Water: In the first experiment, the pre-trained policy is transferred and then fine-tuned within the MMG model under calm-water conditions.

• Fine-Tuning in Wave Conditions—The transferred policy is adapted to the MMG model operating under wave disturbances, enabling the agent to learn disturbance compensation strategies in addition to path-following control.

Developing these two controllers enables a clear comparison of the handling model complexity, and on building resilience to disturbances. The proposed two-stage transfer learning framework and architecture of the actor network are presented in Figure 4. The process begins with a pre-training stage where a DDPG agent learns fundamental control dynamics in a simplified environment. This is followed by a knowledge transfer step, where the learned policy is used to initialize an agent in a high-fidelity environment for a final fine-tuning stage.

The knowledge transfer from Stage 1 to Stage 2 was performed by directly setting the initial weights of the Stage 2 networks to the final, converged weights of the Stage 1 networks.

Let $\boldsymbol{\theta}^*_{A1}$ and $\boldsymbol{\theta}^*_{C1}$ be the final converged weights of the actor and critic networks from the pre-training stage, respectively. The initial weights for the fine-tuning stage ($\boldsymbol{\theta}_{A2,0}$ and $\boldsymbol{\theta}_{C2,0}$) are set as:

$$\begin{cases} \boldsymbol{\theta}_{A2,0} = \boldsymbol{\theta}^*_{A1} \\ \boldsymbol{\theta}_{C2,0} = \boldsymbol{\theta}^*_{C1} \end{cases} \tag{13}$$

This initialization allows the agent in the complex MMG environment to begin fine-tuning from the expert policy

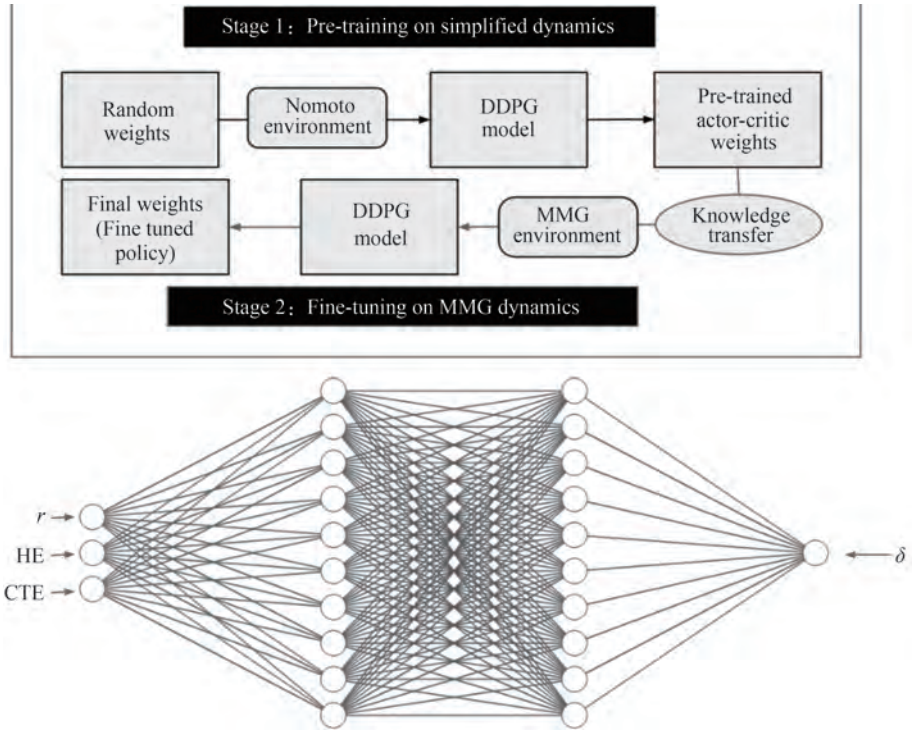


Figure 4 The two-stage transfer learning pipeline for the framework and Architecture of the Neural Network

learned in the simplified Nomoto environment, rather than from random weights.

5.4 PID controller tuning

For the implementation of the PID controller, tuning of the proportional (K_p), integral (K_i), and derivative (K_d) gains are carried out using MATLAB Simulink. Specifically, the Simulink PID Tuner employs an automated frequency-domain loop-shaping algorithm that calculates gains to balance performance and robustness, achieving a target bandwidth (response time) and a stable phase margin.

Initially, the First-Order Nomoto model defined in Equation (3) (Section 2.2) is employed as the linearized reference plant for the PID Tuner. The simplified dynamics $T\dot{r} + r = k\delta$ allow for efficient determination of the initial gains.

Using reference tracking as the criterion, the preliminary tuned parameters were obtained as:

$$K_p = 5.218, K_i = 0.456, K_d = 14.48$$

The gains obtained from the linear Nomoto model served as the starting point for manual tuning on the full nonlinear MMG model. The gains were systematically adjusted by trial and error, running simulations to determine the heading error overshoot and settling time. This iterative process yielded the final tuned gains of

$$K_p = 4.5, K_i = 0.01, K_d = 4$$

which provided stable, responsive performance for the MMG model.

As in the reinforcement learning (RL) -based control experiments, the PID controller is designed to regulate only the heading of the Autonomous Surface Vessel (ASV). The vessel's propeller rotation speed is kept constant throughout the experiments, while the rudder angle is manipulated to minimize heading error and achieve path following.

5.5 Evaluation

The DRL agents developed through our transfer learning framework are benchmarked against two key controllers:

- A traditional PID (Proportional-Integral-Derivative) controller
- DDPG without Transfer Learning Model

The performance of the trained controllers is evaluated using a set of quantitative metrics that assess both tracking accuracy and control efficiency. The evaluation focuses on the following key parameters:

- Heading Error (e_ψ)
- Cross-Track Error (y_e)
- Training Time
- Rudder Stability
- Average Surge Velocity

6 Results

This section presents the empirical results from training and evaluating the DRL controllers developed using our proposed transfer learning framework. The results are presented in two primary subsections. First, the training data is examined, focusing on the agents' learning behavior and convergence. Second, we present the testing results, which include quantitative comparisons of heading error (e_ψ), cross-track error (y_e), and time to convergence of the controller's final performance with that of other methods under various simulated environmental conditions.

6.1 Training

The training performance of the proposed framework is evaluated by comparing the reward convergence behavior of the MMG model under two environmental conditions—calm water and wave disturbances—using three different training strategies:

- Without Transfer-Training MMG model from scratch.
- Transfer from First-Order Nomoto Model-Initializing the MMG model with policy weights learned from a simplified first-order Nomoto model.
- Transfer from Second-Order Nomoto Model-Initializing the MMG model with policy weights learned from the second-order Nomoto model.

Each training run was carried out for 500 000-time steps (approximately 700–800 episodes) with a learning rate of 0.001. On the available hardware, each experiment required roughly 60 minutes of training time. The analysis focuses on the convergence of the training reward over time, which serves as a key indicator of the learning speed and efficiency for each method.

Figure 5 compares the training performance of the MMG model in calm-water conditions under three different training strategies: without transfer learning (teal line), transfer from a First-Order Nomoto (FON) model (orange line), and transfer from a Second-Order Nomoto (SON) model (purple line). The agent initialized using the Second-Order Nomoto (SON) model exhibits exceptional performance from the very beginning, achieving a high episode reward of around 600 within the first 50 episodes. By accurately capturing higher-order dynamics such as phase lag and overshoot, the SON model provides a mathematically sound foundational policy that transfers seamlessly to the complex environment.

In contrast, the agent initialized with the First-Order Nomoto (FON) model experiences a severe initial drop in

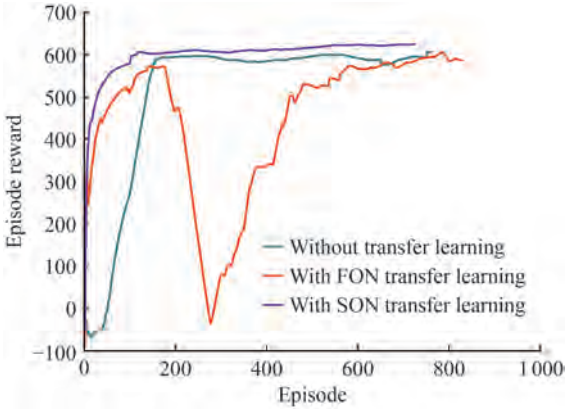


Figure 5 Reward convergence in calm water condition

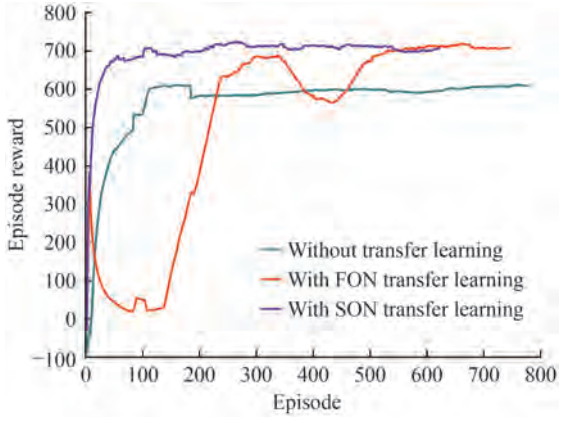


Figure 6 Reward convergence with wave disturbances

performance, plunging into negative rewards before eventually recovering. This phenomenon, known as negative transfer, occurs because the simplified first-order physics, which assume an immediate, proportional yaw response, fundamentally clash with the heavy inertia and sway-yaw coupling of the KVLCC2 tanker in the nonlinear MMG environment. The DDPG agent must first "unlearn" the aggressive steering habits acquired in the FON environment, temporarily causing it to perform worse than the baseline model. Meanwhile, the no-transfer method begins learning from scratch, requiring roughly 200 episodes to approach a stable performance level, reflecting the high sample complexity of learning directly in the MMG environment.

Figure 6 illustrates the training performance of the MMG model under wave disturbance conditions for the same three training strategies. The trends observed in calm water persist in this more challenging, stochastic environment. The SON transfer approach demonstrates a clear and immediate advantage, maintaining high rewards from the outset and proving its robustness against wave drift forces. The FON transfer model again exhibits significant negative transfer due to the compounded complexity of environmental disturbances acting on a mismatched dynamic expectation. It successfully overwrites the deficient policy and converges, but only after extensive exploration. These results highlight the inadequacy of first-order linear approximations for deep reinforcement learning on large displacement vessels, while strongly validating the use of the higher-order SON framework for safe and accelerated knowledge transfer.

These results confirm that initializing the policy with knowledge from a simpler source model enables faster convergence, improved sample efficiency, and more stable early learning, even under environmental uncertainty.

6.2 Testing

The performance of the trained controllers was evaluated in two test scenarios: straight-line trajectory (path length $50\sqrt{2}$ m) in all four quadrants and multiple waypoints (Cardioid Shape) following with a relatively short inter-waypoint spacing of 15 m. The cardioid curve generated based on Eq. (11) represents a closed end trajectory. The variable 'a' in Eq. (11) is chosen as 25.

$$\begin{cases} x(\phi) = 2a(1 - \cos\phi) \cdot \cos\phi \\ y(\phi) = 2a(1 - \cos\phi) \cdot \sin\phi \end{cases} \tag{14}$$

After generating the continuous trajectory in Cartesian coordinates (x, y) for varying ϕ from 0 to 2π radians, waypoints are extracted at approximately 15 m intervals. This is achieved by implementing a waypoint filtering procedure that adds each new waypoint only if its Euclidean distance from the previous waypoint exceeds 15 m, thereby ensuring uniform spacing along the curve.

Table 2 RMSE value for y_e, e_ψ, average rudder angle, and average surge velocity in calm water

Controller	e_ψ (rad)	y_e (m)	Average rudder angle (°)	Average velocity (m/s)
PID	0.250	1.053	29.079	0.480
Model with SON transfer	0.301	1.265	18.231	0.580
Model with FON transfer	0.305	1.342	17.768	0.576
Model without transfer	0.240	1.029	17.145	0.569

6.2.1 Calm water

In the straight-line tracking scenario (Figure 7), all three DRL controllers—without transfer learning, with First-Order Nomoto (FON) transfer, and with Second-Order Nomoto (SON) transfer—can accurately follow the reference trajectories across all quadrants. Notably, the PID controller achieved highly competitive performance, closely following the desired trajectory with minimal deviation. Figure 8

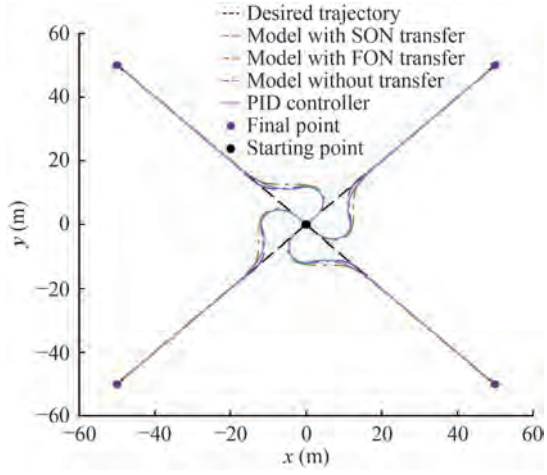


Figure 7 Straight line trajectory in calm water

provides the corresponding rudder angle and surge velocity vs time plots for this scenario. The rudder angles for all models successfully converged to a near-zero value as the vessel stabilized on the straight path. All controllers produced a nearly identical velocity profile.

The second test, shown in Figure 9, presents a more challenging scenario where the agent must navigate a path defined by cardioid shape, with multiple waypoints spaced closely together. All controllers, including the PID and the three DRL variations, performed reliably and efficiently in this demanding situation. Each managed to successfully navigate the complex path, staying near the waypoints despite the steep turns. While the RL-based controllers maintained higher average surge velocities, enabling faster progression along the path, the PID controller exhibited a

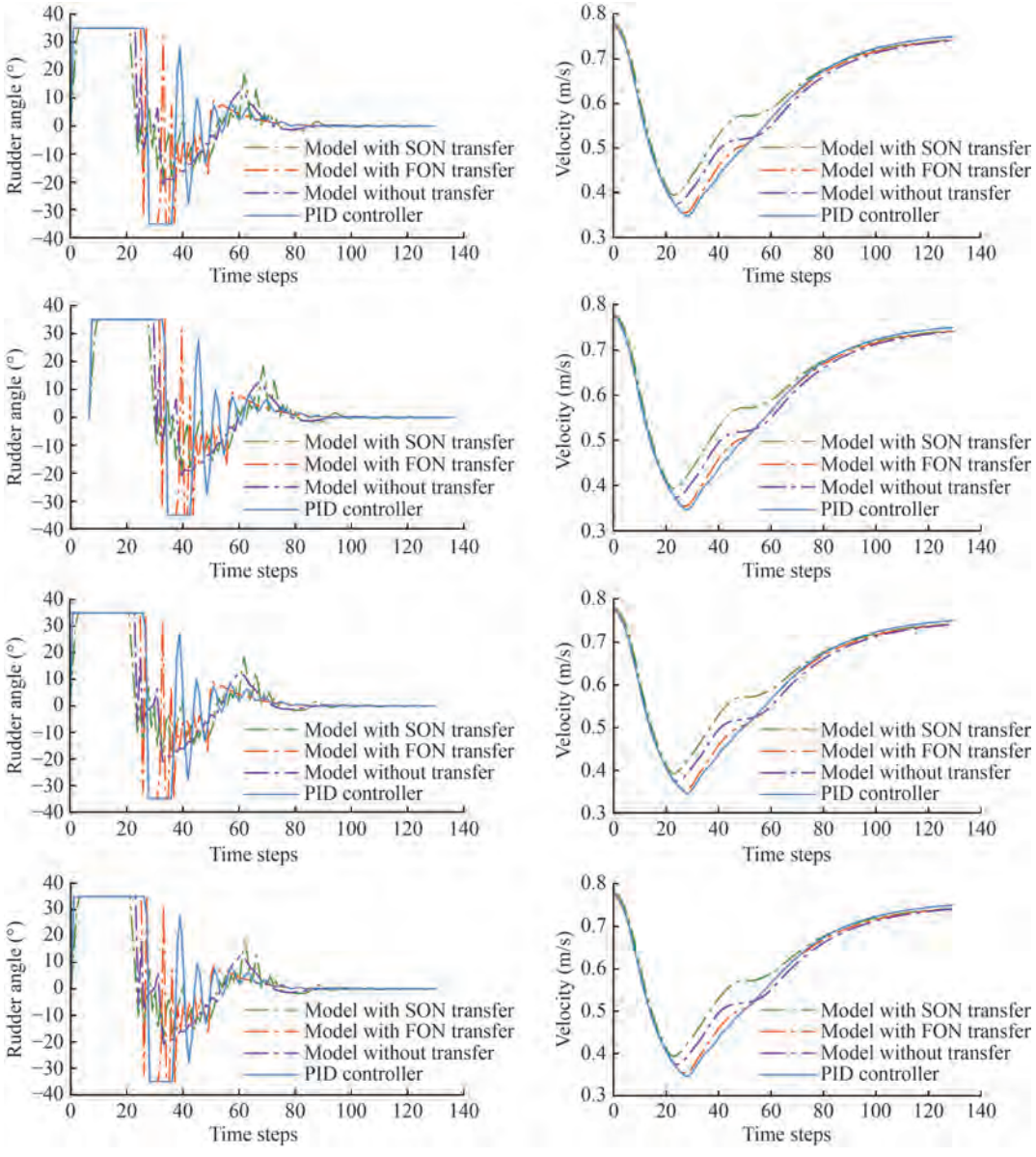


Figure 8 Rudder and surge velocity vs time plot in calm water

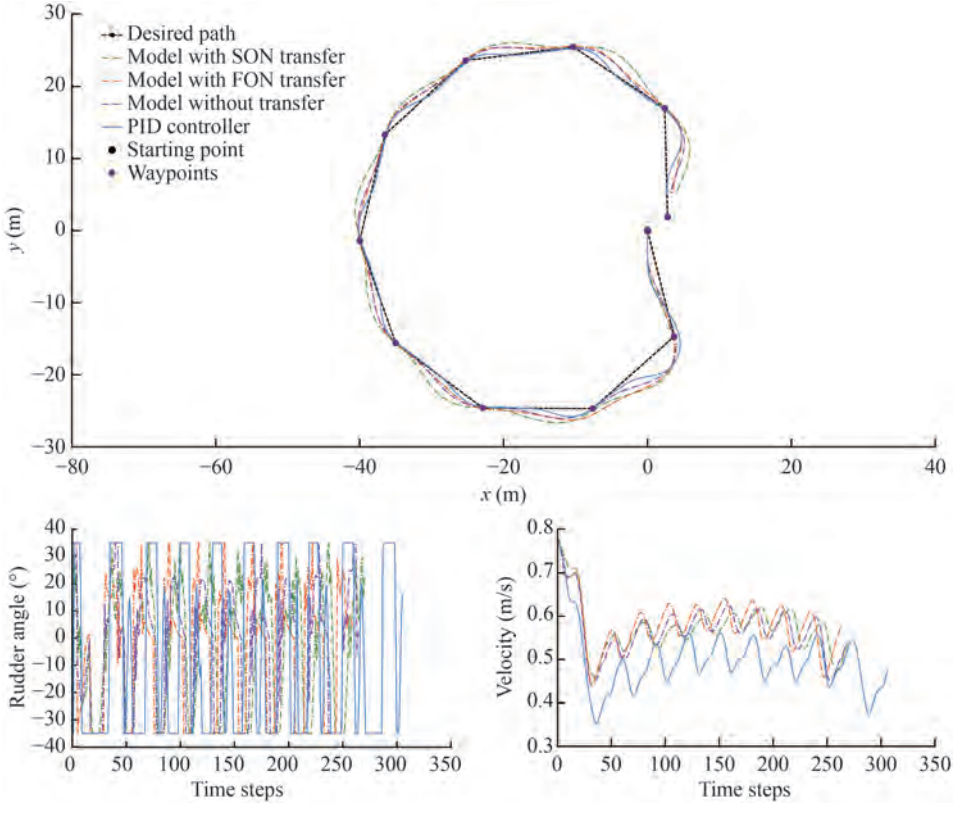


Figure 9 Cardioid trajectory in calm water

steadier rudder response with fewer oscillations, though at the cost of reduced velocity.

6.2.2 Wave disturbances

For the straight-line tracking test (Figure 10), all three controllers—without transfer, with FON transfer, and with SON transfer—successfully maintained their trajectories across all quadrants despite the presence of wave disturbances. The DRL controller trained without transfer achieved the best overall performance, with minimal overshoot and steady-state error, while the SON transfer model showed slightly higher cross-track error. The PID controller also maintained stable behaviour, closely following the desired trajectory with minimal deviation. Figure 11 shows rudder and surge velocity responses indicate that all controllers produced nearly identical dynamic behavior.

In the cardioid shape scenario with wave disturbances (Figure 12), all controllers successfully completed the path, but their performance varied in terms of tracking precision and control effort. As shown in Table 3, the PID controller maintained the lowest cross-track error but at the cost of a higher average rudder angle and lower surge velocity. The SON transfer model achieved the highest surge velocity but also exhibited the largest heading and cross-track errors, indicating reduced robustness in disturbance. The FON transfer and non-transfer models provided a balance between accuracy and efficiency, with moderate cross-track error and lower rudder activity than PID, while maintaining higher average velocities. The rudder and velocity profiles further highlight that DRL-based controllers adapt more dynamically to disturbances, whereas PID emphasizes stability at the expense of speed.

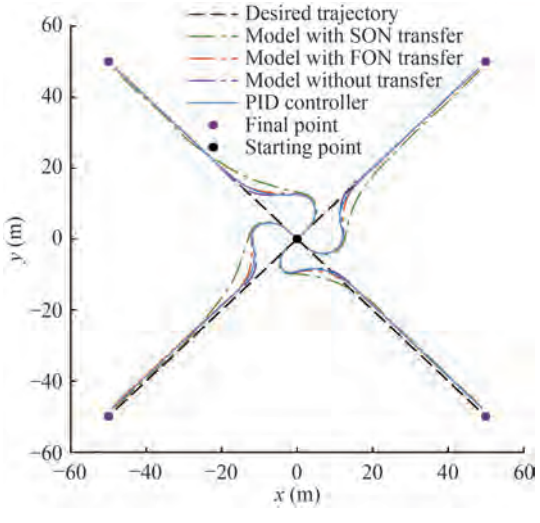


Figure 10 Straight line trajectory with wave

Overall, these results highlight a clear trade-off: while the PID controller ensures stable but slower tracking with higher rudder usage, the DRL-based controllers achieve faster navigation with reduced control effort, though at the cost of slightly higher tracking errors.

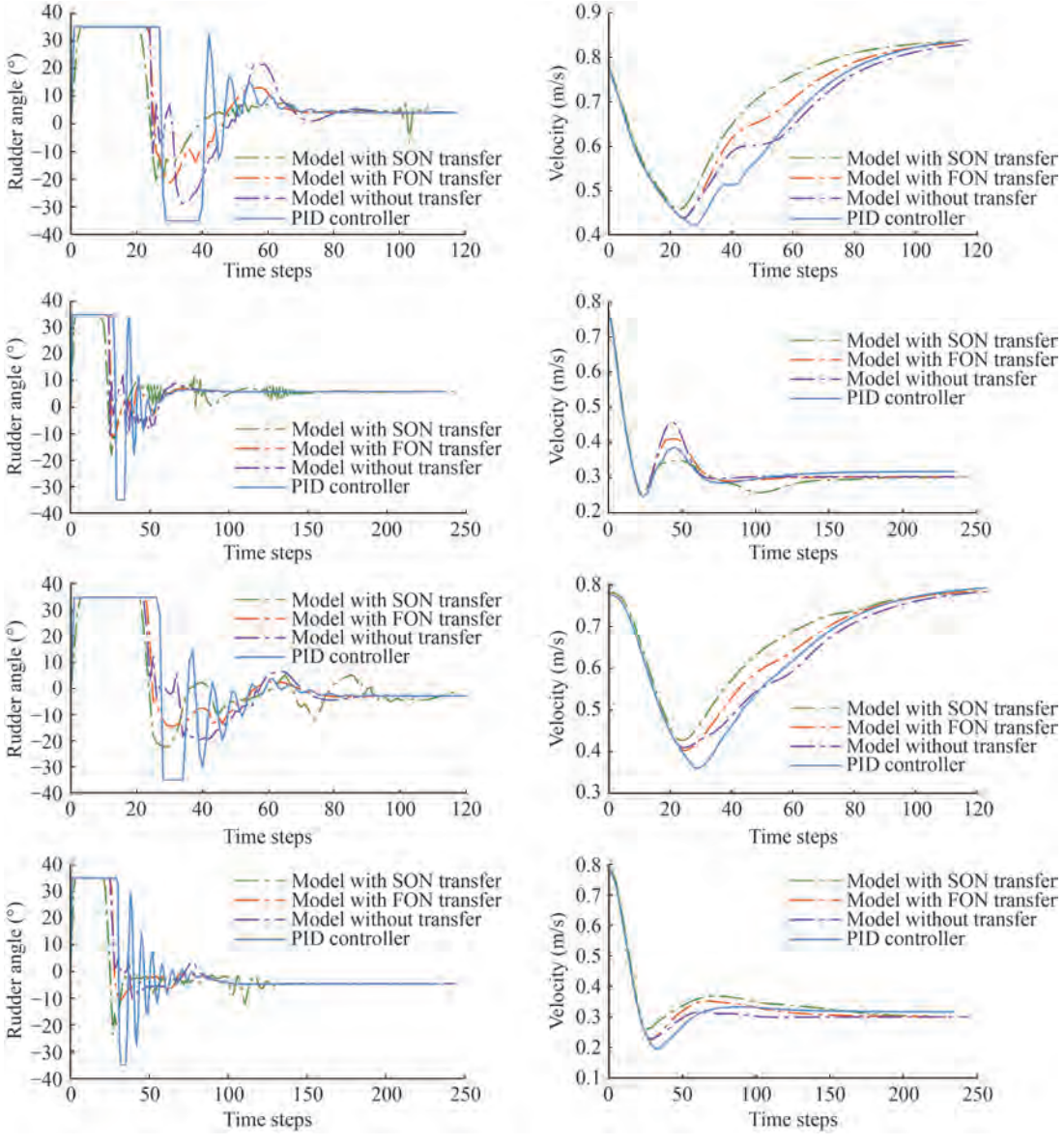
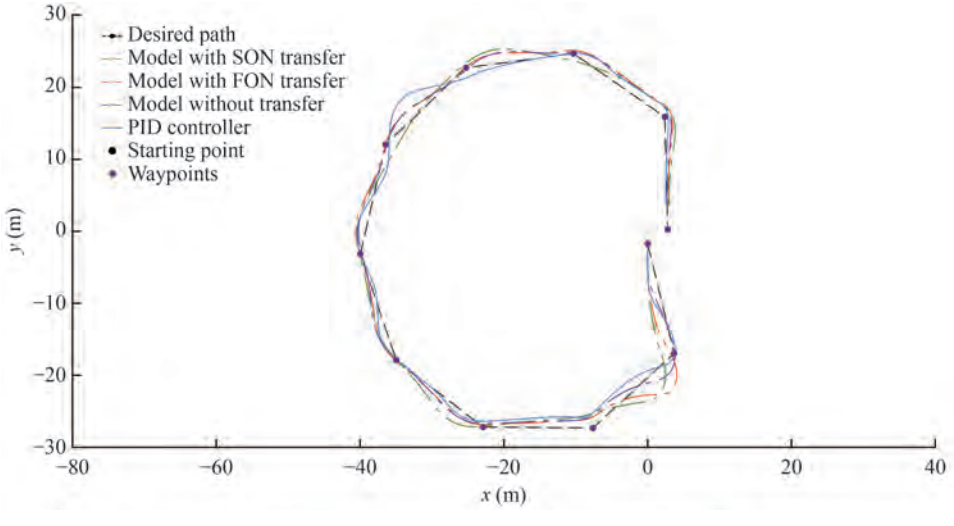


Figure 11 Rudder and surge velocity vs time plot with wave



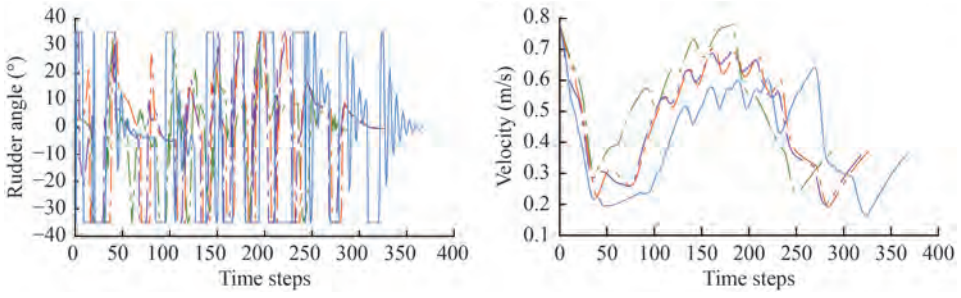


Figure 12 Cardioid trajectory with wave disturbance

Table 3 RMSE value for y_e, e_ψ, average rudder angle, and average surge velocity with wave

Controller	e_ψ (rad)	y_e (m)	Average rudder angle (°)	Average velocity (m/s)
PID	0.287	1.16	22.254	0.398
Model with SON transfer	0.398	1.53	20.958	0.496
Model with FON transfer	0.302	1.33	19.108	0.523
Model without transfer	0.303	1.23	19.773	0.462

7 Conclusions

This work presented a transfer-based deep reinforcement learning (DRL) framework for path-following control of Autonomous Surface Vessels (ASVs), leveraging knowledge transfer from simplified Nomoto models to the nonlinear MMG model of a KVLCC2 vessel. The proposed method accelerated the training process by a factor of four to five compared to training directly in the complex MMG environment, as the policy initialization from simplified dynamics provided a strong starting point.

The results highlight a clear trade-off across controllers. The PID controller provided stable performance with consistent tracking but required higher rudder activity and exhibited lower average surge velocity. In contrast, the DRL-based controllers, particularly the FON transfer and non-transfer models, offered faster navigation and reduced rudder effort, though with slightly higher heading and cross-track errors. The SON transfer model achieved high speed and accuracy but showed reduced robustness to disturbances, as indicated by comparatively higher y_e and e_ψ values.

For future work, the proposed framework will be integrated into ROS 2/Gazebo for high-fidelity simulation and subsequently deployed on a real ASV platform to validate performance in physical trials. The methodology used in this study can be extended to more complex maritime autonomy tasks, such as formation control and collision avoidance for numerous homogeneous or heterogeneous vessels. In addition, future research will investigate feature-level transfer rather than simply transferring network weights between environments, potentially enhancing generalization and adaptability across varying vessel dynamics and operating conditions.

Appendix A: Formulation of the MMG standard method dynamics

A.1 Fundamental assumptions

The mathematical model for the ship's maneuvering predictions is based on the following fundamental assumptions:

- The ship is treated as a rigid body.
- The hydrodynamic forces acting on the ship are treated quasi-steadily.
- The lateral velocity component is assumed to be small compared with the longitudinal velocity component.
- The ship speed is assumed to be sufficiently low such that wave-making effects can be neglected.
- The metacentric height $\overline{GM}$ is sufficiently large, rendering the roll coupling effect on maneuvering negligible

A.2 Total hydrodynamic forces

The total forces and moment on the right-hand side of the motion equations (X, Y, N) are expressed as the superposition of the individual hydrodynamic forces acting on the ship hull (H), the propeller (P), and the rudder (R):

$$\begin{cases} X = X_H + X_R + X_P \\ Y = Y_H + Y_R \\ N = N_H + N_R \end{cases}$$

A.3 Hull hydrodynamic forces

The hydrodynamic forces and yaw moment acting on the bare hull are expressed using non-dimensionalized polynomial functions of the lateral velocity v'_m and yaw rate r'. They are defined as:

$$X_H = (1/2)\,\rho L_{\text{pp}} d U^2 \left[-R'_0 + X'_{vv} v'^2_m + X'_{vr} v'_m r' + X'_{rr} r'^2 + X'_{vvvv} v'^4_m \right]$$

$$Y_H = (1/2)\rho L_{pp} dU^2\left[Y'_v v'^2_m + Y'_r r' + Y'_{vvv} v'^3_m + Y'_{vvr} v'^2_m r' + Y'_{vrr} v'_m r'^2 + Y'_{rrr} r'^3\right]$$

$$N_H = (1/2)\rho L_{pp} dU^2\left[N'_v v'_m + N'_r r' + N'_{vvv} v'^3_m + N'_{vvr} v'^2_m r' + N'_{vrr} v'_m r'^2 + N'_{rrr} r'^3\right]$$

where ρ is water density, L_{pp} is ship length between perpendiculars, d is ship draft, U is the resultant speed, R'_0 is the straight-moving resistance coefficient, and the remaining terms (X'_{vv}, Y'_v, N'_v, etc.) represent the maneuvering hydrodynamic derivatives.

A.4 Propeller forces

The surge force due to the propeller (X_P) is expressed as:

$$X_P = (1 - t_P)T$$

where t_P is the thrust deduction factor. The propeller thrust (T) is defined as:

$$T = \rho n_P^2 D_P^4 K_T(J_P)$$

where n_P is propeller revolution, D_P is propeller diameter, and K_T is the thrust open water characteristic, which is approximated as a second-order polynomial of the advanced ratio (J_P):

$$K_T(J_P) = k_2 J_P^2 + k_1 J_P + k_0$$

$$J_P = \frac{u(1 - w_P)}{n_P D_P}$$

where u is the surge velocity and w_P is the wake coefficient at the propeller position in maneuvering motions.

A.5 Rudder forces and hull-rudder interactions

The effective rudder forces (X_R, Y_R, N_R), which account for the hydrodynamic interactions between the hull and the rudder, are expressed as:

$$\begin{cases} X_R = -(1 - t_R)F_N \sin\delta \\ Y_R = -(1 - a_H)F_N \cos\delta \\ N_R = -(x_R - a_H x_H)F_N \cos\delta \end{cases}$$

where δ is the rudder angle, t_R is the steering resistance deduction factor, a_H is the rudder force increase factor, and x_H represents the longitudinal coordinate of the acting point of the additional lateral force induced by steering.
The rudder normal force (F_N) is calculated by:

$$F_N = (1/2)\rho A_R U_R^2 f_\alpha \sin\alpha_R$$

where A_R is the profile area of the movable part of the rudder, f_α is the rudder lift gradient coefficient, U_R is the resultant rudder inflow velocity, and α_R is the effective inflow angle to the rudder. The kinematics at the rudder are expressed as:

$$\begin{cases} U_R = \sqrt{u_R^2 + v_R^2} \\ \alpha_R = \delta - \tan^{-1}\left(\frac{v_R}{u_R}\right) \end{cases}$$

where the lateral inflow velocity (v_R) accounts for flow straightening (γ_R):

$$v_R = U\gamma_R(\beta - l'_R r')$$

and the longitudinal inflow velocity (u_R) incorporating the propeller slipstream is modeled as:

$$u_R = \epsilon u(1 - w_P)\sqrt{\eta\left\{1 + \kappa\left(\sqrt{1 + \frac{8K_T}{\pi J_P^2}} - 1\right)\right\}^2 + (1 - \eta)}$$

where ϵ is the ratio of wake fraction at the rudder to the propeller, $\eta = (1 - w_R)/(1 - w_P)$, and κ is an experimental constant.

Appendix B: Second-order wave drift force formulation

The steady second-order wave drift forces used in the present study are computed following the formulation of Salvesen (1974). Only the essential resolving equations required for the computation of the steady drift forces are summarized here. The formulation proposed by Salvesen (1974) is based on potential-flow theory with linear first-order wave-body interactions. The total first-order velocity potential is decomposed into incident, diffraction, and radiation components. The body disturbance potential is assumed to be small compared with the incident wave potential (weak-scatterer assumption). The steady second-order drift forces arise from the time-averaged quadratic interaction of the first-order potentials. Three-dimensional ship hydrodynamics are approximated using strip theory, which reduces the problem to a series of two-dimensional sectional solutions along the ship length. The formulation assumes small-amplitude regular waves and constant forward ship speed.

B.1 Mean drift force expression

The steady second-order force obtained from the time-

averaged quadratic interaction of the first-order potentials is expressed as

$$F = -\frac{i\rho k}{2}\int_{S_B}\left[\phi_B\left(\frac{\partial\phi_0^*}{\partial n}\right) - \left(\frac{\partial\phi_B}{\partial n}\right)\phi_0^*\right]\mathrm{ds}$$

where S_B is the wetted body surface, ϕ_0 is the incident-wave potential, ϕ_0^* is the complex conjugate and ϕ_B is the body potential. This is the steady-state force expression obtained by time-averaging the quadratic first-order terms.

$$\phi_B = \sum_{j=1}^{6}\left(\xi_j\phi_j + \phi_7\right)$$

where ϕ_j—radiation potential for the j^{th} mode, ξ_j is the motion amplitude, and ϕ_7 is the diffraction potential

B.2 Decomposition of the drift force

The total steady drift force can be decomposed into three contributions

$$F = \sum_{j=1}^{6}\left(F_j^I + F_j^R\right) + F_7$$

B.3 Havelock contribution

$$F_j^I = \frac{i\rho k\xi_j}{2}\int_{S_B}\left(\frac{\partial\phi_j}{\partial n}\right)\phi_0^*\mathrm{d}S$$

This term represents the interaction between body motions and the Froude–Krylov excitation forces.

B.4 Radiation-incident interaction

$$F_j^R = -\frac{i\rho k\xi_j}{2}\int_{S_B}\phi_j\left(\frac{\partial\phi_0^*}{\partial n}\right)\mathrm{d}S$$

This term represents the interaction between the radiation waves generated by ship motions and the incident wave field.

B.5 Diffraction contribution

$$F_7 = -\frac{i\rho k}{2}\int_{S_B}\phi_7\left(\frac{\partial\phi_0^*}{\partial n}\right)\mathrm{d}S$$

This term represents the interaction between the diffraction potential and the incident wave field.

Competing interests Xueqian Zhou is an editorial board member for the Journal of Marine Science and Application and was not involved in the editorial review, or the decision to publish this article. All authors declare that there are no other competing interests.

References

Busoniu L, De Bruin T, Tolić D, Kober J, Palunko I (2018) Reinforcement learning for control: Performance, stability, and deep approximators. Annual Reviews in Control 46: 8-28. https://doi.org/10.1016/j.arcontrol.2018.09.005

Katebi MR, Moradi MH (2001) Predictive PID controllers. IEE Proceedings-Control Theory and Applications, 148(6): 478-487. https://doi.org/10.1049/ip-cta:20010786

Kiumarsi B, Vamvoudakis KG, Modares H, Lewis FL (2017) Optimal and autonomous control using reinforcement learning: A survey. IEEE transactions on neural networks and learning systems 29(6): 2042-2062. https://doi.org/10.1109/TNNLS.2017.2773458

Lekkas AM, Fossen TI (2013) Line-of-sight guidance for path following of marine vehicles. Advanced in marine robotics 5: 63-92

Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, Wierstra D (2015) Continuous control with deep reinforcement learning. arXiv preprint arXiv: 1509.02971. https://doi.org/10.48550/arXiv.1509.02971

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. arXiv preprint arXiv: 1312.5602. https://doi.org/10.48550/arXiv.1312.5602

Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Hassabis D (2015) Human-level control through deep reinforcement learning. nature 518(7540): 529-533. https://doi.org/10.1038/nature14236

Pan SJ, Yang Q (2009) A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22(10): 1345-1359. https://doi.org/10.1109/TKDE.2009.191

Paramesh S, Rajendran S (2021) A unified seakeeping and manoeuvring model with a PID controller for path following of a KVLCC2 tanker in regular waves. Applied Ocean Research 116: 102860. https://doi.org/10.1016/j.apor.2021.102860

Perera LP, Ferrari V, Santos FP, Hinostroza MA, Soares CG (2014) Experimental evaluations on ship autonomous navigation and collision avoidance by intelligent guidance. IEEE Journal of Oceanic Engineering 40(2): 374-387. https://doi.org/10.1109/JOE.2014.2304793

Puterman ML (2014) Markov decision processes: discrete stochastic dynamic programming. John Wiley and Sons

Salvesen N (1974) Second-Order Steady-State Forces and Moments on Surface Ships in Oblique Regular Waves. DTNSRDC Report 3170. David Taylor Naval Ship Research and Development Center, Bethesda, Maryland

Sandeepkumar R, Rajendran Suresh, Mohan Ranjith, Pascoal, Antonio (2021) A unified ship manoeuvring model with a nonlinear model predictive controller for path following in regular waves. Ocean Engineering. 243. https://doi.org/10.1016/j.oceaneng.2021.110165

Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M (2014) Deterministic policy gradient algorithms. In International conference on machine learning (pp. 387-395). Pmlr https://proceedings.mlr.press/v32/silver14.html

Sivaraj S, Dubey A, Rajendran S (2023) On the performance of different deep reinforcement learning based controllers for the path-following of a ship. Ocean Engineering 286: 115607. https://

doi.org/10.1016/j.oceaneng.2023.115607

Sutton RS, Barto AG (2018) Reinforcement learning: An introduction second edition. Adaptive computation and machine learning: The MIT Press Cambridge MA and London

Taylor ME, Stone P (2009) Transfer learning for reinforcement learning domains: A survey. Journal of Machine Learning Research: 10(7). https://doi.org/10.48550/arXiv.2009.07888

Tomera M (2017) Fuzzy self-tuning PID controller for a ship autopilot. In Marine Navigation (pp. 93-103). CRC Press. https://doi.org/10.1201/9781315099132

Yasukawa H, Yoshimura Y (2015) Introduction of MMG standard method for ship maneuvering predictions. Journal of marine science and technology 20(1): 37-52. https://doi.org/10.1007/s00773-014-0293-y