International Journal of Electrical Engineering and Ethics- Volume 7 Issue 3, 2024

RESEARCH ARTICLE OPEN ACCESS

Ship Navigation System using Deep Reinforcement Learning

Ashis De[1], Barun Mazumdar[2], Sujay Gorain[3], Anup Mondal[4], Sibani Jana[5], Susmita Khata[6]

[1, 2](Department of Electronics & Communication Engineering, Gargi Memorial Institute of Technology, Baruipur, Kolkata
Email: [1]ashis_de@ymail.com, [2]mazumdar.barun684@gmail.com)

[3-6](Department of Electrical Engineering, Gargi Memorial Institute of Technology, Baruipur, Kolkata
Email: [3]sujaygorain789@gmail.com, [4]mondalanup4301@gmail.com, [5]janasibani03@gmail.com,
[6]susmitakhata2003@gmail.com)

Abstract:

A majority of marine accidents that occur can be attributed to errors in human decisions. Through automation, the occurrence of such incidents can be minimized. Therefore, automation in the marine industry has been receiving increased attention in the recent years. This paper investigates the automation of the path following action of a ship. A deep Q-learning approach is proposed to solve the path-following problem of a ship. This method comes under the broader area of deep reinforcement learning (DRL) and is well suited for such tasks, as it can learn to take optimal decisions through sufficient experience. This algorithm also balances the exploration and the exploitation schemes of an agent operating in an environment. A three-degree-of-freedom (3-DOF) dynamic model is adopted to describe the ship's motion. The Krisco container ship (KCS) is chosen for this study as it is a benchmark hull that is used in several studies and its hydrodynamic coefficients are readily available for numerical modeling. Numerical simulations for the turning circle and zig-zag maneuver tests are performed to verify the accuracy of the proposed dynamic model. A reinforcement learning (RL) agent is trained to interact with this numerical model to achieve waypoint tracking. Finally, the proposed approach is investigated not only by numerical simulations but also by model experiments using 1:75.5 scaled model.

Keywords **— Deep Reinforcement learning, Autonomous vessel, Ship maneuvering, Path-following, Deep Q-network, MMG model, Winch control, Active heave compensation.**

I. INTRODUCTION

Over the years, various classical control techniques have been analysed and compared to keep a suspended payload regulated when the vessel is undergoing dynamic vertical motion in the ocean (Zinage and Somayajula (2020); Li et al. (2019); Woodacre et al. (2018); Do and Pan (2008)).

However, not much research has looked at using reinforcement learning (RL) control techniques, which is one of the methods gaining popularity in marine applications (Woo et al. (2019); Martinsen and Lekkas (2018); Zhao and Roh (2019)).

Human error accounts for roughly 80%–85% of all marine-related accidents (Baker and McCafferty, 2005). These incidents put human lives in danger and also have a potential to cause damage to the environment. In addition, the owners of the ships involved in such incidents also face significant financial losses. The recent Ever Given incident in the Suez Canal, 2021 is just one such example, where the vessel could not maintain its path under the influence of strong winds. With the recent advancements in the artificial intelligence (AI) it is now possible to explore automation solutions for the maritime industry to reduce the occurrence of such incidents. The progress in AI technology, particularly reinforcement learning (RL), now offers a new solution to address the demands of ship path following and trajectory tracking.

With the increase in the number of ocean explorations and a huge demand for various marine resources, heave compensation has become a vital part of various maritime operations. In heave compensation, the primary objective is to decouple the motion of a payload connected to the ship from the ship's vertical (heave) motion. Heave compensation methods can be broadly classified into two main categories: passive heave compensation (PHC) and active heave compensation (AHC). The PHC is an open loop system that is designed to partially decouple the

payload from the vessel. The compensation performance for PHC is generally observed to be less than 80 % (Hatleskog and Dunnigan (2007)). AHC relies on closed loop control system architecture and provides the payload displacement as a continuous feedback to the controller so that an improved compensation performance is achieved.

In RL, agents are trained on a reward and penalty system. The agent is rewarded for actions that allow the attainment of a goal and penalized for actions that have detrimental effects. Through experience, the agent seeks the optimal policy that consistently chooses actions leading to higher rewards and avoids actions that lead to lower rewards. Such a control approach falls under the class of data driven controllers where no model of the system being controlled is needed. Rather the system dynamics and the appropriate control strategy is learned by the agent as it tries to optimize its interaction with the system.

Lately, model-free deep reinforcement learning has made significant progress in solving a variety of complex tasks. The first successful application of this technique to learn the control policy was through a deep Q-network (DQN) for playing Atari games (Mnih et al. (2013); Silver et al. (2017)), which integrates the Q learning and deep neural network. However, DQN can only be used to solve problems that have discrete action space. Since manycontrol tasks in the real world have continuous action space, several advanced reinforcement learning algorithms aiming to solve continuous control problems have also been developed (Lillicrap et al. (2015); Mnih et al. (2016); Levine et al. (2016)).

Traditional RL requires the storing of the action choosing policy of the agent in the form of Q-tables. Q-tables document the expected reward for a large table of scenarios covering all possible combinations of discrete states that the agent can be in and the actions it can choose. This table is updated at every time step when the agent moves to a new state through a chosen action from previous state. The advancement in the area of deep learning has given rise to deep reinforcement learning (DRL) that incorporate neural networks to store the policy of the agent (Mnih et al., 2013). This replacement of Q-tables by neural networks allows for a more efficient storage of the policy and potential application to more complex problems. While the traditional RL based on Q-tables requires both state space and action space to be discrete, with deep reinforcement learning it is now possible to explore solutions for scenarios where the state space and action space may be discrete or continuous (Perera et al., 2015).

In this paper, a deep deterministic policy gradient (DDPG) (Lillicrap et al. (2015)) algorithm that is based on an actor critic framework is used. This method has an advantage over the DQN approach that it can deal with a continuous action space. Apart from that, this algorithm inherits conventional approaches of RL such as actor-critic (Sutton et al. (1999)), and policy gradient (Konda and Tsitsiklis (2000)).

II. MODELING OF SHIP MOTION

In this paper, the KRISO container ship (KCS) is chosen for calculating the dynamic motion in the ocean. The Pierson Moskowitz (PM) wave elevation spectrum corresponding to a significant wave height H_S and peak period T_p for a range of frequencies is defined by

$$S(\omega) = \frac{0.31}{2\pi} T_p H_S^2 \left(\frac{\omega T_p}{2\pi}\right)^{-5} \exp\left(\frac{-5}{4}\left(\frac{\omega T_p}{2\pi}\right)^{-4}\right) \quad (1)$$

An irregular wave elevation time history is generated from the above spectrum. The wave elevation is expressed as a sum of N sinusoidal components as shown below

$$\eta(t) = \sum_{i=1}^{N} A_n \cos(\omega_n t + \emptyset_n) \quad (2)$$

In this study, N is taken as 100001 to simulate a 10000s time history with a sampling time of 0.1s. For a simulation of duration T seconds, the frequency increment is given by $\Delta\omega = 2\pi/T$. At each discrete frequency $\omega_n = n\Delta\omega$, the amplitude of the n^{th} wave component is given by

$$A_n = \sqrt{2S(\omega_n)\Delta\omega} \quad (3)$$

The phase ϕ_n of each wave component is sampled from a uniform distribution between $-\pi$ and π (Somayajula (2017)). In order to avoid repeating of the generated signal the frequencies are randomised as shown below

$$(\omega_n)_{new} = (\omega_n)_{old} + \Delta\omega X \quad (4)$$

where X is a random variable following an uniform distribution between −0.5 and 0.5.

The response amplitude operator (RAO) of the KCS container ship is obtained for the heave, roll and pitch modes using MDLHydroD developed by Guha (2016) which is a frequency domain 3D panel method based tool for analysis of wave structure interaction.

Once the RAO is obtained, the response spectrum of the roll, pitch and heave is calculated as shown below

$$S_{response}(\omega) = |H()|^2 S(\omega) \quad (5)$$

where Sresponse is the response spectrum, H(ω) is the RAO and S(ω) is the wave spectrum. Now the input wave elevation time history is decomposed into its frequency components by taking a fast Fourier transform (FFT). The amplitude of the response at a frequency $\omega_k = (k-1)\Delta\omega$ is then obtained by taking the product of RAO at that frequency and the FFT of input wave elevation time history at the same frequency. Finally, inverse fast Fourier transform (IFFT) is used to convert these three degrees of freedom back into time domain.

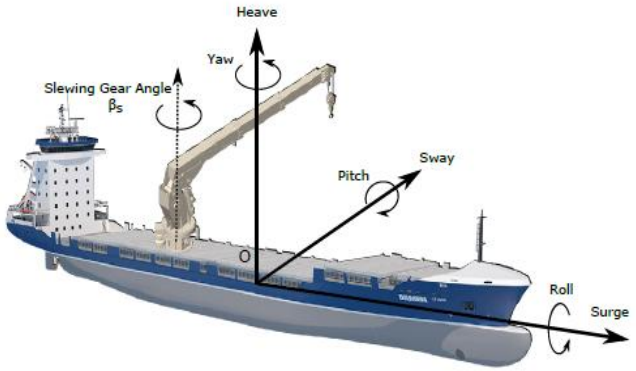


Fig. 1: Ship with Installed Crane

A. Ship Dynamic Model

The KCS vessel is chosen for running numerical simulations and evaluating the control algorithms used in this research. The ship dynamics are mathematically modeled with help of the MMG (ManeuveringModeling Group) model (Yasukawa and Yoshimura, 2015). The 3-DOF non-linear equations of motion are used to solve for ship maneuvering motions including surge, sway and yaw motions. The equations of motion are solved progressively at each time step as an initial value problem using a Runge–Kutta implicit solver. The commanded rudder angle δ_c is provided as an input at each time step.

Two coordinate systems are defined to track the vessel. The first system is a global coordinate system (GCS) that is an earth fixed coordinate frame with its z-axis pointed down. The second is a body coordinate system (BCS) that is fixed to the body and moves with the vessel. The origin of the BCS is located at the intersection of midship, centerline and waterline of the vessel with its x-axis pointed towards the bow, y-axis pointed towards starboard and z-axis pointed towards the keel of the vessel. Both these coordinate frames are shown in Fig. 2.

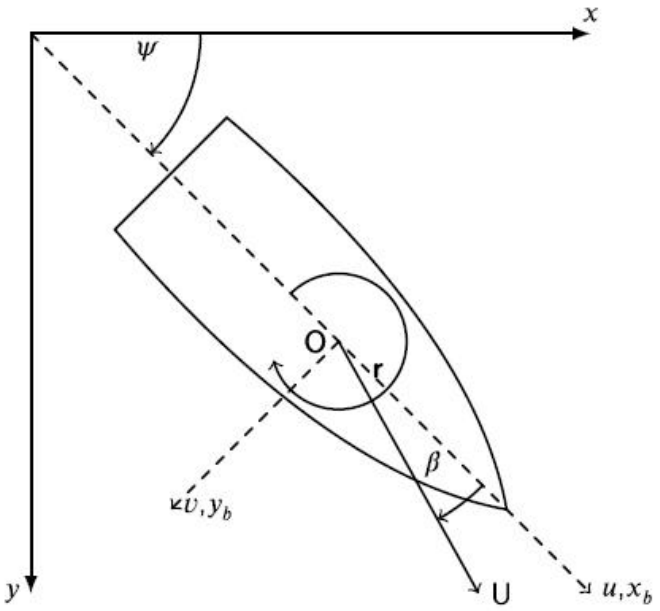


Fig. 2: Representation of ship kinematic variables

The heading angle ψ is defined as the angle between the x-axes of the GCS and BCS frames. The position and orientation of the vessel is denoted by $\boldsymbol{\eta} = [x_0, y_0, \psi]^T$ where x_0 and y_0 denote the position of the origin of BCS expressed in GCS. The velocity vector of the body in BCS is given by $\mathbf{V} = [u, v, r]^T$ where u, v and r represent surge, sway and yaw velocities of the vessel respectively and are expressed in BCS frame. The ship's speed is given by $U = \sqrt{u^2 + v^2}$. The drift angle (β) is defined as the angle between the total velocity vector and the longitudinal direction of ship and is given by $\beta = \tan^{-1}(-v/u)$. The kinematics of ship motion are represented by (6)

$$\dot{\eta} = [R(\psi)]V \quad (6)$$

where $[R(\psi)]$ represents the rotation matrix given by

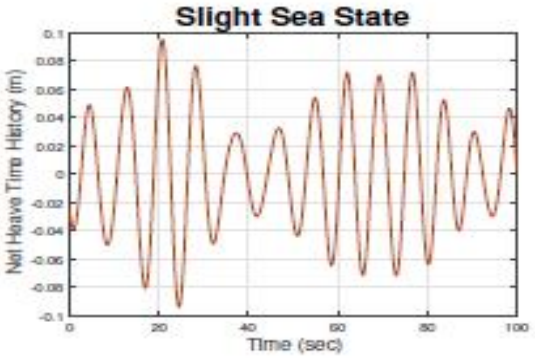
$$[R(\psi)] = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

Any vector in BCS when pre-multiplied by the rotation matrix will result in the same vector expressed in GCS.

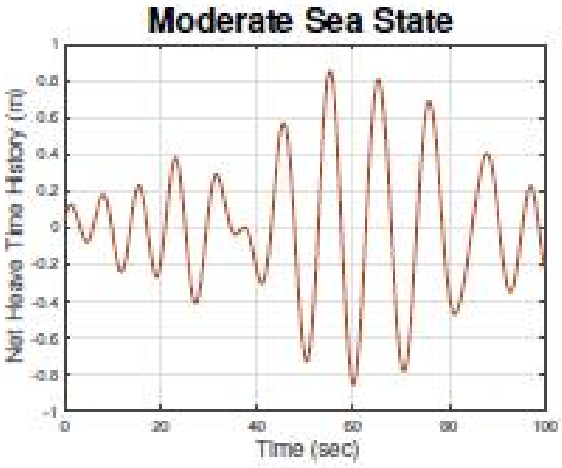
In this study, the origin of the ship fixed coordinate system is assumed to be at the intersection of the waterline, centerline, and midship. Assuming that the crane is placed at (x_{crane}, y_{crane}) in the vessel's body fixed coordinate system with a slewing gear angle β_s and the wave incident angle of β, the net heave response time history of the winch placed on board the KCS ship is calculated in terms of the combined roll, heave, and pitch motion caused due to the wave excitation. Fig. 1 shows a schematic diagram of the ship with a crane installed on it. Assuming small amplitude motions consistent with linear hydrodynamic theory, the net heave motion time history is given by

$$z_{winch} = \eta_3(\beta) + (y_{crane} + l_{crane}\sin(\beta_s))\,\eta_4(\beta) - (x_{crane} + l_{crane}\cos(\beta_s))\,\eta_5(\beta) \quad (8)$$

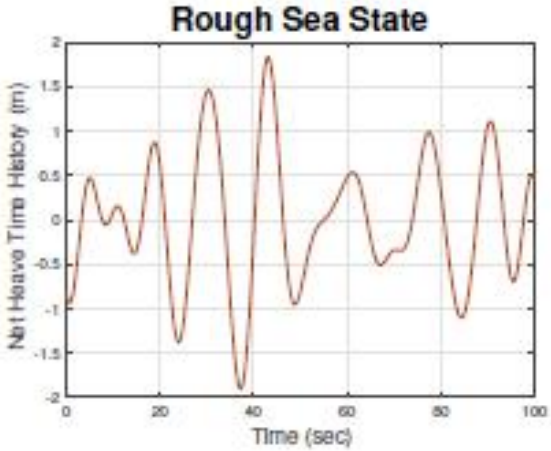
where $\eta_3(\beta)$, $\eta_4(\beta)$, and $\eta_5(\beta)$ are the heave, roll and pitch time histories respectively, which depend on the incident wave angle β. In this study, the coordinate of the crane with respect to vessel's body frame is assumed to be at (−1.5 m, 2 m) with a slewing gear angle of 30 degrees and the horizontal extent of the crane (l_{crane}) is assumed to be 3m. The plot of a 100 second snip of the net heave motion time history in 4 different sea states when the waves are incident at an angle of 135 degrees are shown in Fig. 3.



(a)



(b)



(c)

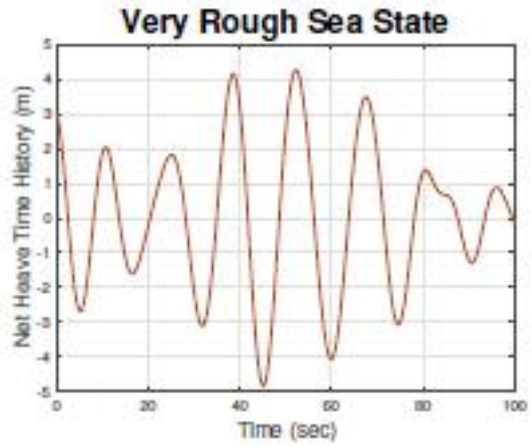


Fig. 3. Net heave time history generated from a PM spectra having (a) Hs = 1.5 m, Tp = 6 s (b) Hs = 4 m, Tp = 9 s (c) Hs = 6 m, Tp = 12 s (d) Hs = 8.5 m, Tp = 14 s

III. REINFORCEMENT LEARNING BASED CONTROLLER

In RL, the agent interacts with the environment by taking actions and observing the state and the reward without any knowledge of the dynamics of the environment. The goal in RL algorithm is to find the optimal policy π^* that selects the optimal control actions u_t^* as shown below

$$u_t^* = \pi^*(s_t) = arg\ max\ Q^{\pi}(s_t, u_t) \quad (9)$$

that maximises the Q value, which is the expected value of the total discounted reward when an action u_t is taken in state s_t. Mathematically, it can be defined as

$$Q^{\pi}(s_t, u_t) = E_{\pi}[R_t|_{s_t, u_t}] \quad (10)$$

where the total discounted reward R_t is given by

$$R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \quad (11)$$

Here, r is the reward obtained at each time step and γ is the discount factor. The discount factor γ determines how much the agent values the reward at the current time step as compared to rewards obtained in the future. The optimal policy π^* is found by using the previous history of the states visited by the agent and the rewards collected by it during its interaction with the environment. By this the agent generates experience which is then used to improve the policy. The action-value function written in a recursive format (also known as the Bellman equation) is given by

$$Q^{\pi}(s_t, u_t) = E_{r_t, s_{t+1}\sim}\varepsilon[r(s_t, u_t) + \gamma E_{u_{t+1}\sim\pi}[Q^{\pi}(s_{t+1}, u_{t+1})]] \quad (12)$$

where the state s_{t+1} is observed from environment ε due to an action u_t selected from state s_t. It is further assumed that the action u_{t+1} is also selected according to the policy π. Since DDPG uses a deterministic policy and the state transition is deterministic under a selected action in this problem, the above equation then becomes

$$Q^{\mu}(s_t, u_t) = r(s_t, u_t) + \gamma Q^{\mu}\left(s_{t+1}, \mu(s_{t+1})\right) \quad (13)$$

where μ represents the deterministic policy function. DDPG adopts an actor-critic framework where both the policy and action-value functions are learnt using neural networks. The actor network takes in the input the current state of the agent and provides the action to be taken according to the policy as its output. The critic network takes in the action and the state as the inputs and provides the Q value as the output. The goal of the critic network is to minimise the mean square temporal difference (TD) error:

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(Q(s_i, u_i|\theta^Q) - y_i\right)^2 \quad (14)$$

where θ^Q represents the parameters of the critic network, N is the sample batch size, and y_i is the temporal difference (TD) target given by

$$y_i = r(s_i, u_i) + \gamma Q^{\mu}\left(s_{i+1}, \mu(s_{i+1})\right) \quad (15)$$

The TD error is defined as difference between the evaluated Q value and the TD target y_i. If y_i is calculated using the same network used by the critic, it may be very hard to converge. So a target critic $Q'\left(s, u|\theta^{Q'}\right)$ and target actor $\mu'\left(s|\theta^{\mu'}\right)$ is introduced. The parameters $\theta^{Q'}$ and $\theta^{\mu'}$ is updated using a soft method given by

$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \quad (16)$$

where $\tau \ll 1$. So, the TD target y_i can be expressed as

$$y_i = r(s_i, u_i) + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1}|\theta^{\mu'}\right)|\theta^{Q'}\right) \quad (17)$$

The aim of the actor network is to maximise the expected accumulated reward J whose gradient is given by

$\nabla_{\theta\mu} J \approx$
$\frac{1}{N}\sum_{i=1}^{N} \nabla_u Q(s,u|\theta^Q)|_{s_i,\, u=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$ (18)
where s_i is sampled from the replay buffer D.

The RL agent is programmed using Keras with a Tensorflow backend. The training has been performed on Nvidia GeForce GTX 1060 16GB GPU. The state space s ∈ S chosen for the application of RL is defined as
$S = \{z_w,\ \dot{z}_w,\ z_{winch},\ \dot{z}_{winch}\}$ (19)

Algorithm 1: DDPG algorithm

1: Randomly initialise critic network $Q(s,u|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
2: Set target parameters equal to main parameters $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$.

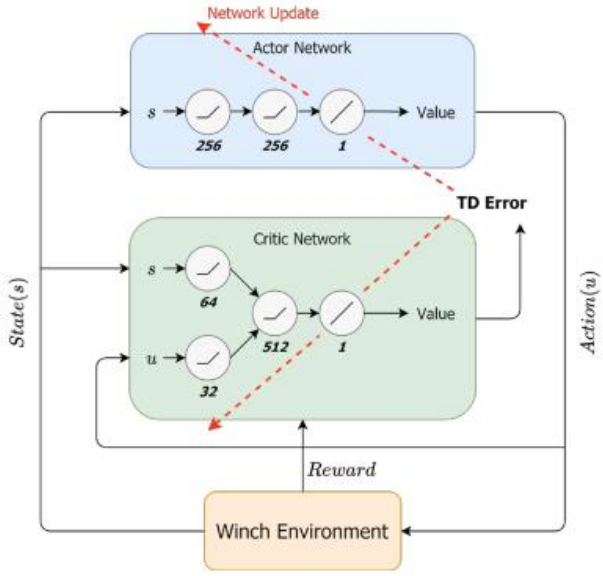


Fig. 4. DDPG Architecture

3: Empty replay buffer D
4: for *episode* = 1, M do
5: Initialise a random noise N for action exploration
6: Receive initial observation state s_0
7: for t = 1, T do
8: Select action $u_t = clip(\mu(s_t|\theta^\mu) + N_t, a_{low}, a_{high})$
9: Execute action u_t in the environment ε
10: Observe new state s_{t+1} and reward r_t
11: Store (s_t, a_t, r_t, s_{t+1}) in D
12: Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from D
13: Compute target $y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1}|\theta^{\mu'}\right)\theta^{Q'}\right)$
14: Update the critic network by minimising the loss $L = \frac{1}{N}\sum_{i=1}^{N}\left(Q(s_i,u_i|\theta^Q) - y_i\right)^2$
15: Update the actor policy by applying the following gradient:
$\nabla_{\theta\mu} J \approx$
$\frac{1}{N}\sum_{i=1}^{N} \nabla_u Q(s,u|\theta^Q)|_{s_i,\, u=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$
16: Update the target networks with
$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$
$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$
17: end for
18: end for

where zw is the length of the reeled rope and zwinch is the net heave at the winch due to the motion of the vessel in waves. The action space u ∈ A is defined as

$$A = \{u_p\} \quad (20)$$

where u_p is the control input provided to the winch model.
Fig. 4 shows a schematic diagram of the structure of the DDPG architecture used. The pseudo code of the DDPG algorithm is shown above. The actor network is composed of two fully connected layers with 256 neurons each whereas the critic network is composed of a fully connected layers with 64 and 32 neurons for inputs s and u respectively followed by a fully connected layer of 512 neurons as shown in Fig. 4. A rectified linear unit function is used as the activation function for each neuron.

IV. SIMULATION RESULTS

The following four cases are analysed to understand the advantages and limitations of using RL based control over classical control.

5.1 *Heave compensation with no disturbance and no noise*

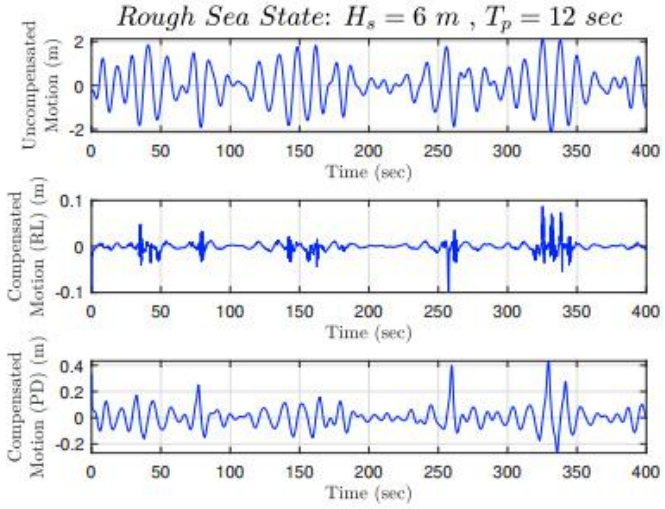


Fig. 5. Uncompensated and compensated motion time histories at the winch for rough sea state.

5.2 *Heave compensation with an offset*

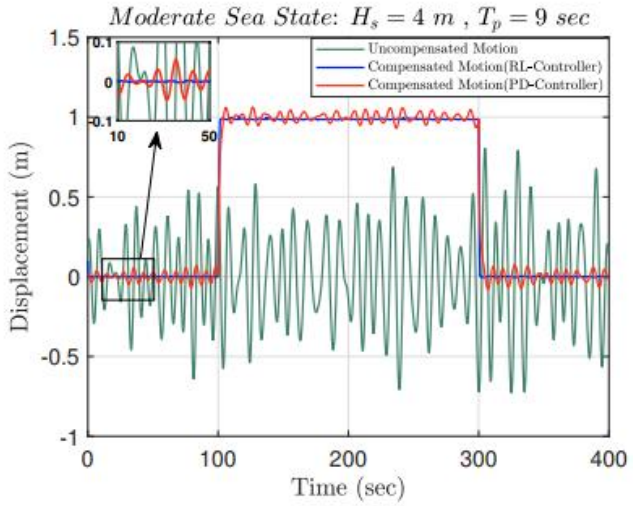


Fig. 6. Ability of the controllers to track the offset

5.3 *Heave compensation with disturbance but no noise*

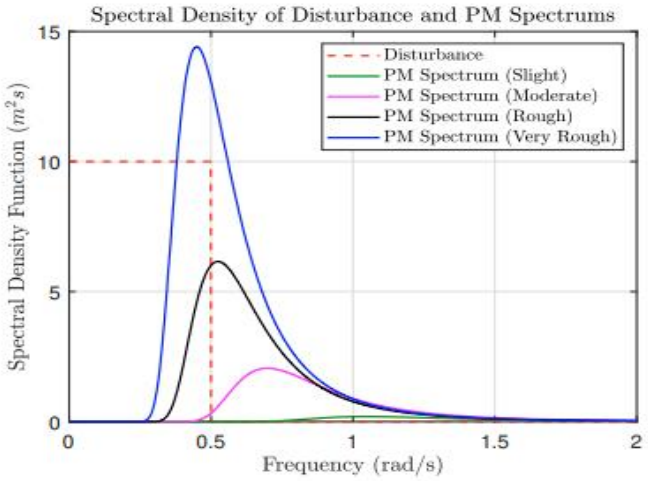


Fig. 7. Spectrum of disturbance and PM spectra's for different sea states

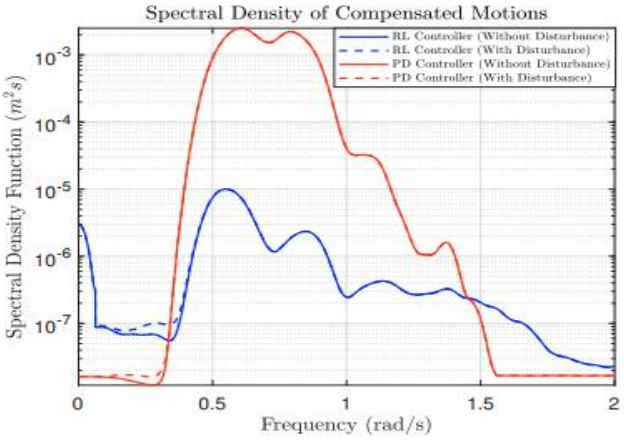


Fig. 8. Spectrum of compensated motions in moderate sea state

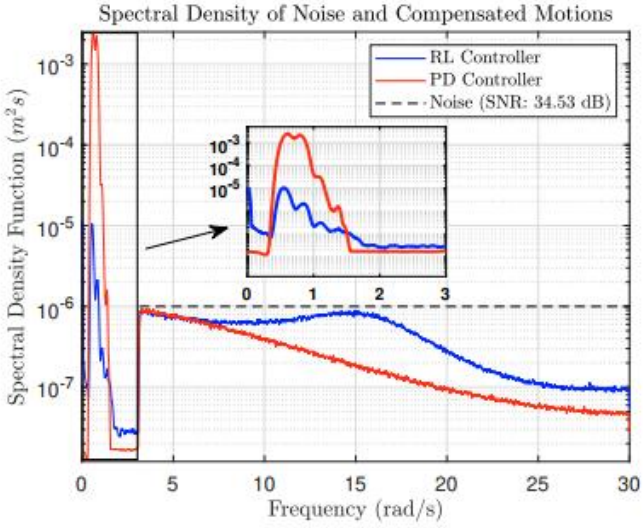


Fig. 9. Spectrum of compensated motions for low noise in moderate sea state

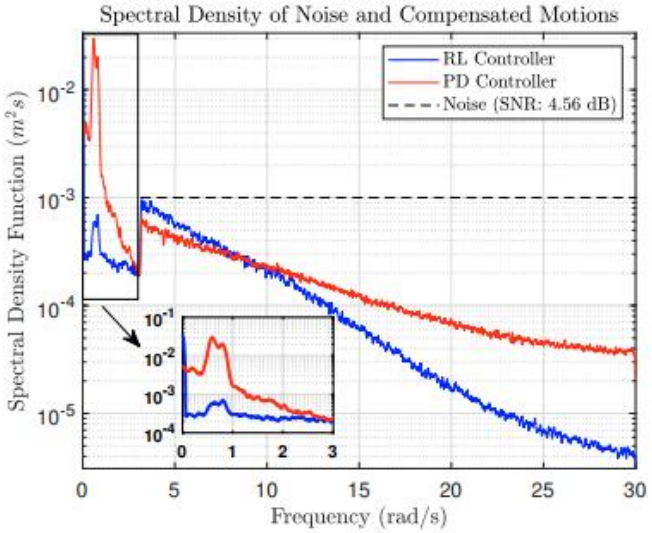


Fig. 10. Spectrum of compensated motions for high noise in moderate sea state

V. CONCLUSIONS

It can be seen that the RL based control is able to demonstrate a better heave compensation performance than the PD control for all the four sea states. Fig. 5 shows the plot of the uncompensated motion and the compensated motion time histories in rough sea state. Fig. 6 shows the ability of the both the controllers in tracking an offset for a period of 200 secs for a wave incident angle of 135 degrees. Fig. 7 shows the plot of the spectrum for the disturbance and the wave elevations in different sea states. In this study, it is assumed that the disturbance entered the system only through the third state (i.e reeled rope velocity ˙ zw). Fig. 8 shows the power spectral density of the compensated motions for both the controllers in the presence and absence of disturbance. It can be observed that both controllers are good at disturbance rejection. Fig. 9 shows the power spectral density of the noise and compensated motions for both the strategies when a low noise is included. From Fig. 9, it can be seen that in high SNR case, the PD controller is able to perform better in attenuating the noise at higher frequencies. As per Fig. 8 and zoomed plot inside Fig. 10, the PD controller is found to be more affected due to the noise as compared to the RL based controller. When it came to high noise attenuation the RL based controller performed better than the PD controller as shown in Fig. 10.

ACKNOWLEDGMENT

First and second authors want to acknowledge Prof. (Dr.) Somnath Maity, Principal, Gargi Memorial Institute of Technology for his constant support and encouragement thrrough the entire work.

REFERENCES

1. *Do, K.D. and Pan, J. (2008). Nonlinear control of an active heave compensation system. Ocean engineering, 35(5-6), 558–571.*
2. *Guha, A. (2016). Development and application of a potential flow computer program: determining first and second order wave forces at zero and forward speed in deep and intermediate water depth. Ph.d. dissertation, Texas A&M University, College Station, TX.*
3. *Hatleskog, J.T. and Dunnigan, M.W. (2007). Passive compensator load variation for deep-water drilling. IEEE Journal of Oceanic engineering, 32(3), 593–602.*
4. *Konda, V.R. and Tsitsiklis, J.N. (2000). Actor-critic algorithms. In Advances in neural information processing systems, 1008–1014. Citeseer.*
5. *Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. The Journal of Machine Learning Research, 17(1), 1334–1373.*
6. *Li, Z., Ma, X., Li, Y., Meng, Q., and Li, J. (2019). Adrcesmpc active heave compensation control strategy for offshore cranes. Ships and Offshore Structures, 1–9.*
7. *Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.*
8. *Martinsen, A.B. and Lekkas, A.M. (2018). Straight-path following for underactuated marine vessels using deep reinforcement learning. IFAC-PapersOnLine, 51(29), 329–334.*
9. *Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In International conference on machine learning, 1928–1937. PMLR.*
10. *Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.*

11. *Richter, M., Schaut, S., Walser, D., Schneider, K., and Sawodny, O. (2017). Experimental validation of an active heave compensation system: estimation, prediction and control. Control Engineering Practice, 66, 1–12.*

12. *Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. nature, 550(7676), 354–359.*

13. *S. Zinage, A. Somayajula, Deep Reinforcement Learning Based Controller for Active Heave Compensation, ScienceDirect, IFAC(2021) 161–167.*

14. *R. Deraj, R.S. Sanjeev Kumar, Md Shadab Alam, A. Somayajula, Deep reinforcement learning based controller for ship navigation, ELSEVIER Ocean Engineering 273 (2023) 113937*

15. *Cui, R., Yang, C., Li, Y., Sharma, S., 2017. Adaptive neural network control of AUVs with control input nonlinearities using reinforcement learning. IEEE Trans. Syst. Man Cybern. Syst. 47 (6), 1019–1029.*

16. *Fossen, T.I., 1999. Guidance and control of ocean vehicles. (Doctors Thesis). University of Trondheim, Norway, Printed By John Wiley & Sons, Chichester, England, ISBN:0 471 94113 1.*

17. *Fossen, T.I., 2021. Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley & Sons.*

18. *Guo, S., Zhang, X., Zheng, Y., Du, Y., 2020. An autonomous path planning model for unmanned ships based on deep reinforcement learning. Sensors 20 (2), 426.*

Heiberg, A., Larsen, T.N., Meyer, E., Rasheed, A., San, O., Varagnolo, D., 2022. Riskbased implementation of COLREGs for autonomous surface vehicles using deep reinforcement learning. Neural Netw. 152, 17–33.