Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 10033–10038

Risk-based Convolutional Perception Models for Collision Avoidance in Autonomous Marine Surface Vessels using Deep Reinforcement Learning

Thomas Nakken Larsen * **Hannah Hansen** * **Adil Rasheed** *

* *Department of Engineering Cybernetics, Norwegian University of Science and Technology, O. S. Bragstads plass 2, Trondheim, NO-7034, Norway (e-mail: thomas.n.larsen@ntnu.no, hannhan@stud.ntnu.no, adil.rasheed@ntnu.no)*

Abstract: In this work, we propose a novel policy network architecture for model-free Reinforcement Learning (RL)-based path-following and collision avoidance in marine surface vessels. By applying convolutional neural networks (CNNs) for mapping LiDAR-like distance measurements to Collision Risk Indices (CRIs), we evaluate the utility of risk-based pretraining of CNN feature extractors prior to RL. Where previous works required hand-crafted preprocessing of high-resolution distance measurements to train an autonomous RL agent successfully, the proposed approach achieves this goal in a data-driven fashion. Ultimately, we propose future directions to improve CNN-based perception models for collision avoidance in range sensing applications.

Keywords: Reinforcement learning and deep learning in control; Autonomous surface vehicles; Neural networks; Collision Risk Indices

1. INTRODUCTION

Human error is a leading cause of accidents on the road (Dingus et al. (2016); Thomas et al. (2013)), and reports show that accidents at sea are no different. According to the Annual Overview of Marine Casualties and Incidents published by the European Maritime Safety Agency (EMSA), human error was attributed to over 50% of accidental events between 2011-17 (EMSA, 2021). In addition to reducing accidents (and thereby fatalities), environmental damage, and costs, autonomous marine operations allow for optimized route planning.

There is an increasing trend in applying deep reinforcement learning (DRL) algorithms for local mission planning and collision avoidance (Sarhadi et al., 2022). To enable collision avoidance in autonomous vessels, the controller must perceive its immediate surroundings. Such measurements will typically be of high dimension and require dimensionality reduction for machine learning controllers. Previous works (Larsen et al., 2021; Heiberg et al., 2022) on reinforcement learning (RL)-based autonomous control for path-following and collision avoidance in marine surface vessels considered a hand-crafted feasibility pooling algorithm for dimensionality reduction. While this approach is highly efficient in encoding the observation space, it also entails a significant information loss.

In this work, we propose using a Convolutional Neural Network (CNN) as an alternative approach for feature extraction. CNNs are widely adopted for feature extraction in high-dimensional, spatially structured data. Wang utilize a VGG-19 CNN pretrained from image classification to extract features in an analogous image generation task. RGB images are typically structured as a $[0,1]^{m\times n\times 3}$ matrix with m and n typically in the order of $10^1 - 10^3$. We consider a LiDAR-like sensor that produces a 1D array of planar distance measurements on the form $[0,1]^N$, where each element covers a $2\pi/N$ angle area (see Fig. 1). We do not consider a typical point cloud output but rather a rangefinder similar to a discretized 2D (planar) LiDAR. One may argue, purely based on the difference in dimensionality, that our LiDAR-like data contains significantly simpler features than natural images. However, LiDAR data has a similar spatial property, i.e., neighboring measurements are highly correlated, given sufficient sensor resolution. Since CNNs efficiently extract localized features from spatial information, they can be useful for encoding the distance and localization of nearby obstacles from high-resolution distance measurements. The CNN-based approach may enable the identification of complex features utilizing the sensor's full resolution compared to the feasibility pooling method.

Moreover, a CNN feature extractor can be seamlessly integrated into a larger neural network structure, e.g., the policy network of an RL agent. In contrast to feasibility pooling, the CNN approach requires learning relevant features in a data-driven fashion. In this work, we investigate the feasibility of pretraining CNNs via supervised regression for collision risk estimation, i.e., finding a mapping from encoded LiDAR data to collision risk prior to RL. To this end, the current research addresses the following questions:

- Can we use a CNN to better utilize the full resolution of the rangefinder sensor?

10.1016/j.ifacol.2023.10.870

- Can we do a reasonably good estimation of risk using CNN?
- Can risk-based CNN pretraining accelerate RL training?

The article is organized as follows: Section 2 presents the relevant theory required to appreciate the work, followed by Section 3, which provides all the information that makes the current work reproducible. Results and discussions are presented in Section 4, followed by a brief conclusion and proposed future work in Section 5.

2. THEORY

2.1 Ship model

Cybership II (Skjetne et al., 2004) is a 1:70 replica of a supply ship, whose dynamics model was estimated through experimental data, given the following assumptions; (1) the vessel moves on a surface, i.e., there is no heave motion; (2) there are no disturbances (wind, currents, or waves) to the vessel. Under these assumptions, the dynamics model is expressed in 3-DOF as

$$\dot{\eta} = \mathbf{R}_{z,\psi}(\eta)\nu \tag{1}$$

$$\mathbf{M}\dot{\nu} + \mathbf{C}(\nu)\nu + \mathbf{D}(\nu)\nu = \mathbf{B}\mathbf{f}, \tag{2}$$

where $\mathbf{R}_{z,\psi}$ represents a rotation of ψ radians about the z-axis; $\eta = [x^n, y^n, \psi]^T$ describes the vessel's position and heading in the North-East-Down (NED) inertial frame; $\nu = [u, v, r]^T$ describes the vessel's translational and angular velocities; $\mathbf{B}$ is the actuator configuration matrix; and $\mathbf{M}$, $\mathbf{C}(\nu)$, and $\mathbf{D}(\nu)$ are the mass, Coriolis, and damping matrices, respectively. Finally, the control input, $\mathbf{f} = [T_u, T_r]$, consists of the surge force and yaw moment. The original model includes a bow thruster which is omitted in this work due to its limited effectiveness at high speeds (Sørensen et al., 2017).

2.2 Collision risk

The collision risk index (CRI) is a common property to quantify collision risk at sea. The CRI combines risk factors to evaluate the risk of collision between the own ship and a target ship, where the distance to the closest point of approach (DCPA) and time to the closest point of approach (TCPA) are preferred factors to include (Gang et al., 2016; Zhao et al., 2016).

Evaluating the risk factors of the CRI is not straightforward, as collision risk assessment is characterized by subjectivity and ambiguity. A common approach is to use fuzzy theory to integrate expert knowledge in the risk estimation (Abebe et al., 2021). Fuzzy comprehensive evaluation methods are among the strategies, where membership functions $u(\cdot) \in [0, 1]$ are used to determine the risk level associated with each risk factor, the final CRI being the weighted sum of the membership functions. Considering the risk factors DCPA, TCPA, relative distance R, relative velocity V and relative bearing θ_{T} as suggested by (Heiberg et al., 2022), the CRI takes the form

$$\begin{aligned}\mathrm{CRI} = \alpha_{\mathrm{CPA}} \sqrt{u(\mathrm{DCPA}) \cdot u(\mathrm{TCPA})} \\ + \alpha_{\theta_{\mathrm{T}}} u(\theta_{\mathrm{T}}) + \alpha_{\mathrm{R}} u(R) + \alpha_{\mathrm{V}} u(V),\end{aligned}$$

where $\alpha_{\mathrm{DCPA}} + \alpha_{\mathrm{TCPA}} + \alpha_{\theta_{\mathrm{T}}} + \alpha_{\mathrm{R}} + \alpha_{\mathrm{V}} = 1$. Further details of the membership functions and numerical values are found in (Heiberg et al., 2022).

2.3 Reinforcement learning

RL is a massive multi-disciplinary field that describes a framework for iterative learning through trial and error. Model-free RL considers an *agent* that learns a decision-making policy through acting in an *environment*, given only scalar reward feedback. Formally, this can be defined as finding the policy parameters, θ^*, that maximize the expected return, i.e., $\theta^* = \arg\max_\theta \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[r(\tau)]$, where τ is the sequence of states and actions, $\{(s_1, a_1), ..., (s_T, a_T)\}$, $\pi_\theta(\tau)$ is the trajectory distribution given by θ, and $r(\tau) = \sum_t r(s_t, a_t)$ is the reward sum over the trajectory (Sutton and Barto, 2018). Model-free RL methods can largely be grouped into three distinct groups: policy gradient, actor-critic, and value-based methods. In this work, we consider the *Proximal Policy Optimization* (PPO) algorithm (Schulman et al., 2017) — a widely adopted policy gradient approach based around advantage estimation in a surrogate policy-improvement objective: $J(\theta') - J(\theta) = \mathbb{E}_{\tau\sim\pi_{\theta'(\tau)}}[\sum_t \gamma^t A^{\pi_\theta}(s_t, a_t)]$. Baker et al. (2020) apply the PPO algorithm on a competitive multi-agent task, where 1D convolution of a LiDAR-like perception model is part of the observation space.

3. METHOD AND SETUP

3.1 Simulation environment

This work considers the simulation framework for autonomous surface vessels in maritime environments, "gym-auv", developed by Meyer et al. (2020b). This OpenAI Gym-style (Brockman et al., 2016) simulator can be used for training RL agents in various marine environments and manually maneuvering the surface vessel. We consider an environment that models a calm ocean surface, including a vessel governed by the 3-DOF Cybership II dynamics model, initialized at the start of a randomly generated path of fixed length. A set of static and dynamic obstacles represent landmasses and other moving vessels. These are distributed on and along the path according to a normal distribution, with the mean on the path. Typically, one would not set the desired path to intersect with known obstacles; we consider an unreliable desired path affected by human error or unknown obstacles to challenge the RL agent's collision avoidance capability. The moving obstacles travel in straight lines at randomly chosen velocities between 0.06 ms^{-1} and 1.99 ms^{-1}. The number of generated static and dynamic obstacles are fixed at 11 and 17, respectively. An example scenario is shown in Fig. 3.

In addition to observing guidance-theoretic navigation features for path-following, the ownship is equipped with a synthetic 2D LiDAR-like sensor suite for distance measurements. A single scan consists of 180 measured distances uniformly distributed in the horizontal plane around the vessel with a range of $150m$, as illustrated in Fig. 1. These distance measurements are transformed into closeness by flipping and normalizing the measurements, such that 0 corresponds to no measured obstacles and 1 to a collision.

3.2 CNN architecture design

Compared to typical deep CNNs applied in computer vision tasks, the networks proposed here are shallow, and the kernels are wide. Fig. 1 illustrates how the 1D convolutional filter interfaces with the sensor data and corresponds

to the minimal CNN architecture (1conv) configuration. The intuition behind this design choice is based on the relative simplicity of the information to be extracted; the distance and relative position of any nearby obstacles should be sufficient to guide an autonomous controller to avoid a collision. While more advanced and modern network architectures exist (e.g., recurrent networks and transformers), we aim to find the simplest representation necessary to solve the task. Table 1 describes the proposed architectures in detail. The investigation of these three configurations serves as an initial, small-scale hyperparameter- and neural architecture search. A more extensive search is considered outside the scope of this work and will be considered in future work.

Table 1. CNN architectures investigated in this work. D, K, S, P, F, and O denote the depth, kernel size, stride, padding, number of features, and output size, respectively. The "Deep" model contains max-pooling layers after each activation. The padding mode (where applicable) is "circular", as illustrated in Fig. 1.

Model	D	K	S	P	F	O
1conv	1	45	15	15	1	12
3conv	3	45,3,3	15,1,1	15,0,0	3,2,1	12
Deep	4	9	1	4	2,4,4,6	8

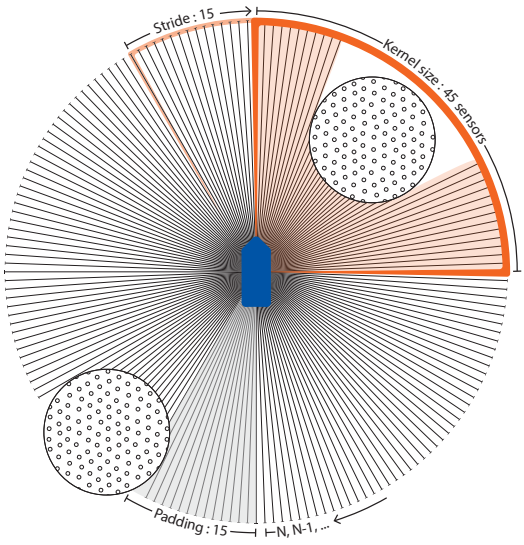


Fig. 1. Mechanism of a convolutional filter processing radial distance measurements. N distance measurements are uniformly distributed in a 1D array starting directly behind the vessel, rotating 2π rad around the vessel. The convolution kernel covers $\frac{N}{4}$ measurements and strides across $\frac{N}{12}$ sensors for each feature activation. Circular padding at the boundaries ensures continuous overlap. This configuration thus encodes N measurements to 12 latent features.

3.3 Mapping LiDAR data to collision risk

The general regression problem can be formulated as $y = \mathcal{F}(X, \theta)$, where $\mathcal{F}$ is a mapping from input X to output y, parameterized by θ. The objective is to find the mapping that minimizes the sum of mean squared error (MSE): $\sum_{i=0}^{N}(\mathcal{F}(X_i, \theta) - y_i)^2$. In this work, we formulate a risk regression problem where distance measurements are mapped to CRIs, using a CNN to parameterize $\mathcal{F}$. The dataset was collected using the simulation framework for autonomous surface vessels in maritime environments by Meyer et al. (2020b). The collision risk associated with each obstacle within range of the sensor is calculated as defined in Heiberg et al. (2022). As multiple CRIs may exist for each scan (one for each nearby obstacle), we define the target risk y to be the maximum CRI present. Thus, given the closeness measurement X, the CNN is trained to predict the maximum associated collision risk y.

3.4 Reinforcement Learning

The PPO algorithm has previously been shown to solve the simultaneous path-following and collision avoidance problem in the aforementioned simulation environment and outperforms a set of other model-free RL algorithms (Larsen et al., 2021). In this work, we consider the same RL control problem and expand the perception model for the RL agent from feasibility pooling to utilizing raw sensor measurements. Meyer et al. (2020a) experienced unstable training when passing the rangefinder measurements directly into a fully connected network. Although a CNN architecture results in fewer learnable parameters and preserves the spatial information of the sensor data, we attempt initializing the CNN subnetwork via supervised risk regression to stabilize and accelerate the RL agent's training. Fig. 2 illustrates the proposed policy network architecture. To evaluate whether pretraining the CNN is beneficial to the RL agent, we further compare the following configurations; 1) the pretrained CNN feature extractor is initialized with locked parameters and will not be modified by the RL algorithm ("*locked*"), 2) the pretrained CNN is initialized, and its parameters are allowed to be modified by the RL algorithm ("*unlocked*"), and 3) the CNN is initialized with random weights and is trained purely through the RL algorithm ("*random*").

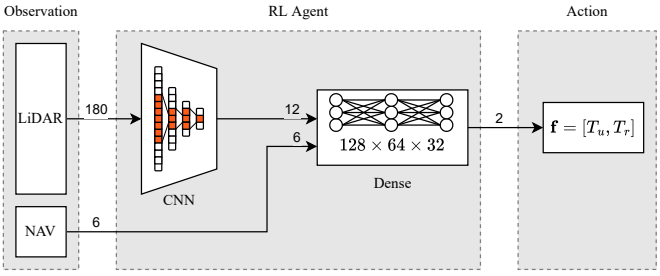


Fig. 2. RL agent architecture. A CNN encodes the LiDAR-like distance measurements, and navigation features are calculated using guidance theory. The concatenated range and navigation features are fed into a fully connected network. The RL agent then outputs the thrust action to act on its environment.

3.5 Experiment setup

This work involves two experiments. Firstly, we collect a dataset of range measurements and CRI pairs to train CNNs via regression. Secondly, we train a variety of DRL agents to solve the path-following and collision avoidance problem. To initialize the CNN subnetwork in the RL agent's policy network, we use the pretrained CNNs obtained in the first experiment.

Pretraining CNNs: As the models are trained to solve a scalar risk regression problem, fully connected regression

heads are appended to the CNN architectures. The regression heads for the CNNs with one and three layers consist of a single linear layer, while the deep network has two hidden layers of size 40 and 8. The data set of 10 000 samples is split into a training, validation, and test set of ratios 0.5, 0.2, and 0.3, respectively. The models are trained to minimize MSE over ten epochs using the Adam optimizer (Kingma and Ba, 2014) with learning rates of $\lambda = 0.0025$, $\lambda = 0.002$ and $\lambda = 0.0005$ for the 1conv CNN, 3conv CNN and DeepCNN, respectively. We store intermediate models and select the ones that produce the lowest validation losses. Finally, the CNNs are evaluated on 100 trajectories in the simulation environment.

RL setup and performance evaluation: We train an RL agent using Stable-Baselines3's PPO implementation (Raffin et al., 2021) for simultaneous path-following and collision avoidance in the synthetic and stochastic environment for each architecture and configuration of CNN feature extractors. The hyperparameters for the PPO algorithm are identical to the ones applied in Larsen et al. (2021). After training the agents for 3M timesteps each, we evaluate whether (1) training an autonomous agent is possible by processing LiDAR-like data with a CNN architecture in the RL agent and (2) pretraining the CNN for risk regression is beneficial for the RL training process. To this end, we evaluate a set of performance attributes over 100 testing episodes — average "progress": how far along a set path the agent can progress, "collision avoidance" (COLAV): how often the agent refrains from colliding, average "cross-track error" (CTE): how far the agent deviates from the set path; and average time spent per episode.

4. RESULTS AND DISCUSSIONS

4.1 CNN risk estimation

MSEs of the predicted risk after training the proposed architectures are displayed in Table 2. All models have a small MSE, indicating that the range data contains sufficient information to estimate the maximum CRI, even though the problem is under-determined; our measurements contain information about the distance and bearing of target ships, while the CRI also depends on their relative velocities. Of the three architectures, the deep CNN yields the lowest MSE by a small margin. This marginal improvement implies that increasing the model complexity yields diminishing returns for risk estimation. The predicted risk along the trajectory by the pretrained 1conv CNN is displayed in Fig. 3. Though there are some deviations between the predicted and true risk, the prediction reasonably follows the true curve.

Table 2. Mean squared error on risk predictions over 100 randomly generated environments

Model	MSE ± ΔMSE
1conv	0.0217 ± 0.0150
3conv	0.0236 ± 0.0141
Deep	0.0156 ± 0.00875

4.2 RL agents

Table 3 presents statistics from testing the RL agents in 100 generated environments. The 1conv and 3conv CNNs

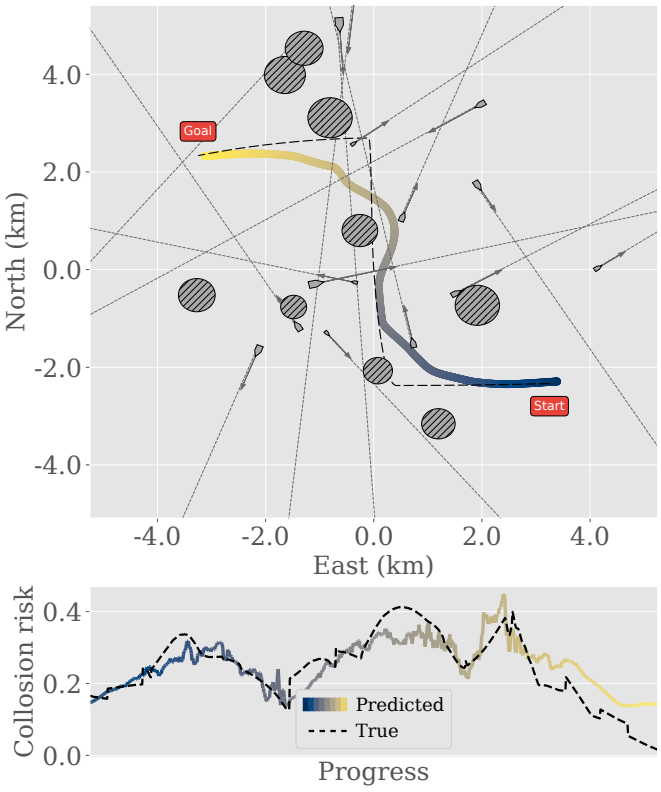
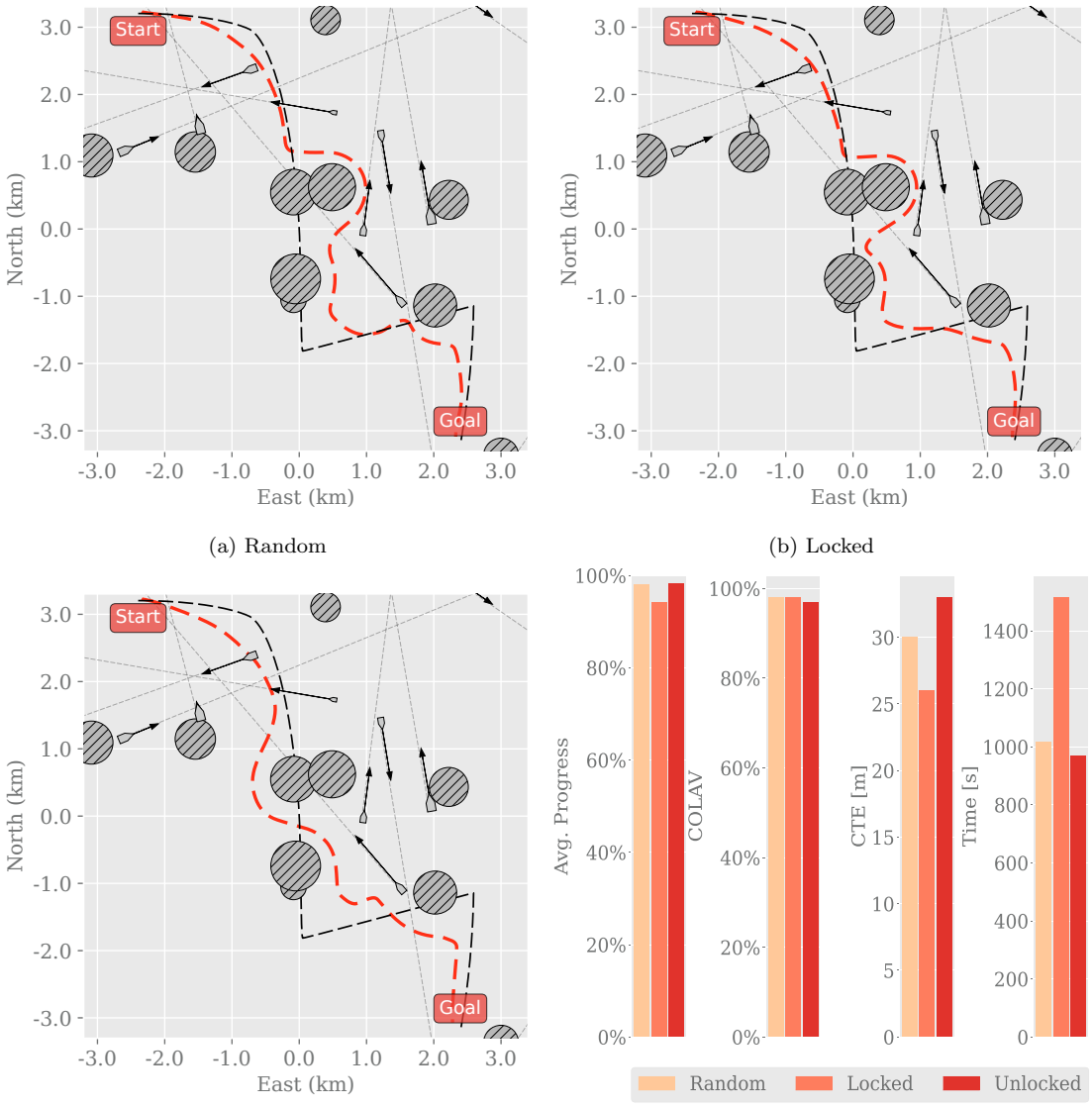


Fig. 3. Performance of an RL-agent with a pretrained 1conv feature extractor with unlocked variables. The upper image shows the environment and the desired path marked by the dashed black line. The colored line is the path taken by the agent. The lower plot shows the risk along the path. The dashed line is the calculated risk, and the colored line is the predicted risk by the pretrained CNN. The color corresponds to the progress along the path.

achieved sufficient average progress to solve the environment, while the deeper CNNs did not. Single-layer CNNs yield low CTE, indicating consistent path adherence but are not collision-free. In contrast, the three-layer CNN yields the best collision avoidance but deviates more from the path. Common for both of the shallow configurations is that locked CNN parameters led to worse performance. Fig. 4 illustrates these results for the 1conv architecture. Finally, deeper CNN had high collision avoidance scores but failed to adhere to the paths and reach the goals. This failure could originate from max-pooling the feature activations, which inherently removes some positional information. Yet, the deep CNN configurations display the most significant impact from pretraining the CNNs via risk regression. In the other configurations, random initializations perform at least as well as the pretrained ones. In summary, where the previous works experienced difficulties in directly utilizing the full resolution of the LiDAR-like data, the CNN-based architecture enables end-to-end learning of an RL agent for path-following and collision avoidance.

5. CONCLUSION AND FUTURE WORK

Our results can be summarized as follows:



(a) Random

(b) Locked

(c) Unlocked

(d) Evaluation metrics.

Fig. 4. Example episode for each configuration of the 1conv CNN DRL agent: (a) randomly initialized parameters, (b) pretrained with locked parameters, and (c) pretrained with unlocked parameters. The black line is the desired path, and the red line is the agent's path. Each configuration's evaluation metrics (d) are averaged over 100 episodes.

Table 3. RL test results averaged over 100 episodes in the simulation environment for each CNN architecture and parameter configuration. Bold entries indicate the best performance in the category.

Configuration	Avg.prog	COLAV	CTE	Time
1conv random	98.03%	98%	$30m$	$1018s$
1conv locked	94.16%	98%	$26m$	$1515s$
1conv unlocked	**98.28%**	97%	$33m$	**971s**
3conv random	97.06%	**100%**	$182m$	$1694s$
3conv locked	93.45%	90%	**20m**	$1010s$
3conv unlocked	94.19%	99%	$113m$	$1894s$
Deep random	44.75%	95%	$774m$	$8278s$
Deep locked	71.93%	90%	$71m$	$3210s$
Deep unlocked	85.28%	100%	$471m$	$3347s$

- We found CNN-based mappings from LiDAR-like distance measurements to CRIs with low MSE, despite the underdetermined problem.
- We trained autonomous path-following and collision avoidance RL agents using guidance-theoretic navigation features for path-following and CNN-based feature extraction to encode a high-fidelity rangefinder sensor for collision avoidance.
- With the proposed CNN architectures, we found that the utility of pretraining the CNNs to accelerate the training of RL agents is limited. However, our results indicate that the utility increases for deeper CNNs rather than shallow ones.

Ultimately, the results show that the shallow CNN-based perception model can yield autonomous path-following and collision avoidance agents that perform comparably to the previous approach using feasibility pooling. While the new approach has room for improvement in collision avoidance, its advantage lies in the ease of implementation on physical hardware. The CNN-based model needs only the distance measurements, whereas the feasibility pooling

method requires additional knowledge of the decomposed velocity components of each nearby obstacle. Other CNN architectures can be investigated to improve the approach, and stacking temporal frames of measurements can enable the CNN to infer the velocity and acceleration of surrounding obstacles.

ACKNOWLEDGEMENTS

This work is part of SFI AutoShip, an 8-year research-based innovation center. We want to thank our partners, including the Research Council of Norway, under project number 309230.

REFERENCES

Abebe, M., Noh, Y., Seo, C., Kim, D., and Lee, I. (2021). Developing a Ship Collision Risk Index estimation model based on Dempster-Shafer theory. *Applied Ocean Research*, 113, 102735. doi:10.1016/j.apor.2021.102735.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2020). Emergent Tool Use From Multi-Agent Autocurricula. In *International Conference on Learning Representations*.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

Dingus, T.A., Guo, F., Lee, S., Antin, J.F., Perez, M., Buchanan-King, M., and Hankey, J. (2016). Driver crash risk factors and prevalence evaluation using naturalistic driving data. *Proceedings of the National Academy of Sciences*, 113(10), 2636–2641. doi:10.1073/pnas.1513271113. Publisher: Proceedings of the National Academy of Sciences.

EMSA (2021). Annual Overview of Marine Casualties and Incidents.

Gang, L., Wang, Y., Sun, Y., Zhou, L., and Zhang, M. (2016). Estimation of vessel collision risk index based on support vector machine. *Advances in Mechanical Engineering*, 8(11), 1687814016671250. doi:10.1177/1687814016671250. Publisher: SAGE Publications.

Heiberg, A., Larsen, T.N., Meyer, E., Rasheed, A., San, O., and Varagnolo, D. (2022). Risk-based implementation of COLREGs for autonomous surface vehicles using deep reinforcement learning. *Neural Networks*, 152, 17–33. doi:10.1016/j.neunet.2022.04.008.

Kingma, D. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*.

Larsen, T.N., Teigen, H.Ø., Laache, T., Varagnolo, D., and Rasheed, A. (2021). Comparing Deep Reinforcement Learning Algorithms' Ability to Safely Navigate Challenging Waters. *Frontiers in Robotics and AI*, 8.

Meyer, E., Heiberg, A., Rasheed, A., and San, O. (2020a). COLREG-Compliant Collision Avoidance for Unmanned Surface Vehicle Using Deep Reinforcement Learning. *IEEE Access*, 8, 165344–165364. doi:10.1109/ACCESS.2020.3022600. Conference Name: IEEE Access.

Meyer, E., Robinson, H., Rasheed, A., and San, O. (2020b). Taming an Autonomous Surface Vehicle for Path Following and Collision Avoidance Using Deep Reinforcement Learning. *IEEE Access*, 8, 41466–41481. doi:10.1109/ACCESS.2020.2976586. Conference Name: IEEE Access.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268), 1–8.

Sarhadi, P., Naeem, W., and Athanasopoulos, N. (2022). A Survey of Recent Machine Learning Solutions for Ship Collision Avoidance and Mission Planning, to be published.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal Policy Optimization Algorithms. doi:10.48550/arXiv.1707.06347. ArXiv:1707.06347 [cs].

Skjetne, R., Smogeli, Ø., and Fossen, T.I. (2004). Modeling, identification, and adaptive maneuvering of CyberShip II: A complete design with experiments. *IFAC Proceedings Volumes*, 37(10), 203–208. doi:10.1016/S1474-6670(17)31732-9.

Sutton, R. and Barto, A. (2018). *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press.

Sørensen, M.E.N., Breivik, M., and Eriksen, B.O.H. (2017). A ship heading and speed control concept inherently satisfying actuator constraints. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*, 323–330. doi:10.1109/CCTA.2017.8062483.

Thomas, P., Morris, A., Talbot, R., and Fagerlind, H. (2013). Identifying the causes of road crashes in Europe. *Annals of Advances in Automotive Medicine*, 57, 13–22.

Zhao, Y., Li, W., and Shi, P. (2016). A real-time collision avoidance learning system for Unmanned Surface Vessels. *Neurocomputing*, 182, 255–266. doi:10.1016/j.neucom.2015.12.028.