Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 54-16 (2021) 7–15

Combining Supervised Learning and Digital Twin for Autonomous Path-planning *

Chanjei Vasanthan * Dong T. Nguyen **

* *Department of Marine Technology, Norwegian University of Science and Technology, NTNU Trondheim, NO-7491 Norway (e-mail: chanjeiv@stud.ntnu.no)*

** *Centre for Autonomous Marine Operations and System (AMOS), Department of Marine Technology, Norwegian University of Science and Technology, NTNU Trondheim, NO-7491 Norway (e-mail: dong.t.nguyen@ntnu.no)*

Abstract: Over the last decade, the evolution of autonomous automobiles based on artificial intelligence has increased rapidly with significant success. Naturally, this has caught the interest of the maritime industry and the development of autonomous vessels. However, unlike the highway, the ocean is considered a complex environment carrying unpredictable environmental forces, such as current, waves and wind-condition. For autonomous path-following and path-planning, particularly within the machine learning-field, Deep Reinforcement Learning (DRL) have generally been the favored approach. This follows from the fact that resulting models have demonstrated staggering performance. However, for practical implementations, Deep learning-based models are generally considered black box-solutions, and hence often introduce uncertainties in the operating domain. Therefore, in this paper an autonomous path-planner based on Supervised learning is proposed. Different Supervised learning models were investigated, and Gradient Boosting Regressor was found to be the most adequate model based on hyperparameter-tuning. The model was developed on constraints proposed by the class society DNV GL combined with International Regulations for Preventing Collision at Sea (COLREGs) rule 14 for collision-avoidance. Following this, the model was trained to design a suitable path based on parametrization of a cubic Bézier curve. To follow the parametrized path, a maneuvering-controller derived from the Maneuvering problem presented in Skjetne (2005) was applied. However, a drawback of Supervised learning is the necessity for large-scale training data. Hence, a digital twin of the own vessel was developed and utilized to generate sufficient training data. To demonstrate the performance of the autonomous path-planner, a number of simulation scenarios were introduced.

Keywords: Autonomy, Path-planning, Supervised learning, Digital twin, Maneuvering, Collision avoidance, COLREGs

1. INTRODUCTION

Following the recent advancement of computational power combined with the rise of machine learning, the popularity of the latter has increased tremendously. This has naturally led to an extensive research on new potential domains of application. In this article we will consider one such domain, namely autonomous vessels. In general, we define an autonomous system as a system capable of decision-making without human interference (Sørensen, 2018). A well-known example is the self-driving car, which has demonstrated the ability to autonomously maneuver in traffic. This has unsurprisingly stimulated the research within the maritime industry as well. One such outcome is the upcoming zero-emission autonomous container ship Yara Birkeland, currently under development by the Kongsberg Group (Kongsberg Group, 2020). Similar to self-driving cars, autonomous vessels aim to possess the ability to navigate the sea without human interference. However, compared to the highway, the ocean can be classified as a significantly more complex environment. Especially when considering environmental variables such as current, wind, waves and surrounding vessels. In fact, the potentially fatal consequence of such a difficult task was recently demonstrated by the accident of the Royal Norwegian Navy vessel, Helge Ingstad (Stangvik et al., 2019). Hence, it is certain that an autonomous vessel also demands high level of intelligence and re-planning abilities when unforeseen scenarios occur.

Traditionally, Model Predictive Control (MPC) have been the preferred approach for automatic vessel maneuvering, especially in terms of practical application. Hence, a substantial amount of research naturally revolves around further development of existing solutions. For instance, Blindheim et al. (2020) showcased a MPC strategy for autonomous ship in terms of emergency management.

* This work was sponsored by the Research Council of Norway through the Centre of Excellence funding scheme, project number 223254, AMOS.

10.1016/j.ifacol.2021.10.066

However, including collision avoidance (COLAV) naturally introduce uncertainty to an already complex calculation, as discussed in Luman et al. (2019). Especially since the optimal path has to be calculated online. Despite that, promising early-stage solutions utilizing MPC have been presented such as in Eriksen and Breivik (2017) and Zheng et al. (2014). Additionally, there have also been extensive research on adaption of motion planning-algorithms that have been successfully implemented for unmanned ground vehicles. For instance, in Singh et al. (2017) a study applying the well-established Djikstra algorithm for path-planning during static environment was successfully made. Similarly, a noticeable approach based on Rapidly-exploring Random Tree* (RRT*) was presented in Zaccone (2021) with satisfying results for medium-range and short-range collision avoidance system.

However, within the machine learning-field, Deep learning-based solutions have mainly been the favored approach during development of autonomous vessels, more specifically Deep Reinforcement learning (DRL). This is not unexpected considering the noteworthy performance demonstrated in Bell and Lekkas (2018) using Deep Deterministic Gradient Policy (DDPG). In this paper, a DRL-controller was implemented to learn to follow curved paths during unknown environment. However, to accomplish higher level of autonomy, obstacle-avoidance is essential and considered a critical task. In Vallestad (2019) an extension of the previous solution was proposed incorporating the rules for maneuvering beside a head-on vessel, but with limited success. Although the proposed solution made the vessel capable of avoiding collision, the vessel incorrectly maneuvered on the port-side of the head-on vessel instead of the starboard-side. Additionally, the resulting control outputs were considerable noisy, similar to Bang-Bang control-behavior. A corresponding study applying DDPG-controller to command the rudder angle was proposed in Aronsen (2019). Despite the controller initially guiding the vessel towards the endpoint, it eventually drifted off without any reasonable explanation. Consequently, these examples demonstrate some of the limitations related to Deep learning-based solutions, namely the level of explainability and uncertainty related to the Deep learning models. As discussed in Buhrmester et al. (2019) and Dulac-Arnold et al. (2019), Deep learning models are often regarded as a black box. Thus, understanding the decision process behind an action is generally considered difficult as demonstrated by for example AlphaGo. AlphaGo is a computer program trained with Deep Reinforcement learning to play the board-game Go (Li and Du, 2018). In 2016, it was capable of beating the best human player at the time, Lee Sedol, in a five-game match. By making unexpected moves, unthinkable even for the experts, the computer became victorious. However, in terms of applying new technologies in the public society, it is expected that the solution holds the ability to safeguard both the environment and human lives while operating. Hence, despite demonstrating astonishing results, with respect to safety, the possibility of an intelligent program making unexpected actions may not be favorable, even if the actions are convenient at the moment. Especially considering the resulting uncertainties that the model introduces in the operating environment. Similar issues have been discussed extensively in the medical care where human lives are at stake, such as in Kelly et al. (2019) and Ghassemi et al. (2020). As a result, the application of Deep learning in the medical field has mainly been focused on image processing, where the consequences of the uncertainties related to Deep learning solutions are considered small.

The objective of this paper is to propose a solution based on Supervised learning for autonomous path-planning. In contrast to DRL, Supervised models provide high level of transparency and explainability. Hence, understanding the process behind the prediction or decision-making is achievable. However, a major drawback of Supervised learning is the necessity for a large amount of data to train the model. Especially generating and collecting data from a real vessel is normally considered a difficult task. Therefore, by developing an adequate simulator of a real vessel, we implement the concept of digital twin to be able to generate sufficient data for the training process (Mendi et al., 2021). Note that in the previously mentioned studies, DRL was mainly implemented as a controller trained to maneuver the vessel from A to B, which often resulted in noisy control output on the thrusters. However, these days the performance of traditional controllers for path-following are considered more than satisfactory. Hence, a traditional controller is used in combination with a Supervised model. In particular, the Supervised model is trained to generate paths, such that the controller can calculate appropriate control commands to maneuver the vessel along the path. To model the paths, we utilize the mathematical parametrization of cubic Bézier curves. Further, to assure stability during path-following, we implement a maneuvering model based on the Maneuvering problem proposed in Skjetne (2005). An important aspect during any successful technological development is considering the viewpoints of the authority and experts within the industry. Hence, additional constrains are introduced based on the criteria presented by the class society for an autonomous path-planner. Finally, to demonstrate the capability of the proposed solution, we try to solve the head-on vessel situation based on International Regulations for Preventing Collision at Sea (COLREGs) rule 14.

The paper is composed of five sections. Section 2 presents background on the legal expectations of an autonomous vessel path-planner, definition of COLREGs rule 14, vessel modelling, path-following, Bézier curves and Supervised learning. Section 3 outlines the implementation stages of the solution, followed by a presentation and discussion of the simulation results in section 4. Finally, in section 5 a conclusion of the paper is made.

2. PRELIMINARIES

2.1 *Autonomous path-planning: A class society perspective*

For the maritime industry the rules and regulations are mainly governed by the International Maritime Organization (IMO), an organization part of the United Nations. To maintain the safety and security of the industry, IMO outlines conventions and legal instruments, such as the International Convention for the Safety of Life at Sea (SOLAS) and International Regulations for Preventing Collision at

Sea (COLREGs). The class societies on other hand, in particular Den Norske Veritas Germanischer-Lloyd (DNV GL) and Lloyds Register aim to legally verify that the design, construction and maintenance of the vessels satisfy the necessary standards. Each class society maintains their own set of class rules that cover the technical requirement related to each component of a vessel. At present time, IMO has not yet outlined any specific regulation for novel technologies like autonomous vessels. However, as stated in DNV GL (2018), national- and regulatory associations can support the implementation of such solutions within their local territorial waters. As a consequence, DNV GL has outlined a guidance for development of autonomous solutions in DNV GL (2018). Note that a similar, but less detailed guidance has been provided by Lloyds Register in Lloyd's Register (2017).

Generally, the navigation task can be divided into four sub-tasks: condition detection, condition analysis, action planning and action control. Each task can be performed by a human, a system or both. However, it is expected that any system introduced need to be as good as, or better than the conventional solution in order to maintain equivalent level of safety. An example is provided in fig. 1. Here it can be noted that condition detection, action planning and action control are made by the system, while condition analysis is partly performed by the human operator. The objective of this paper is to introduce a self-controlling (SC) system for action planning, also known as an autonomous path-planner. To solve the action planning

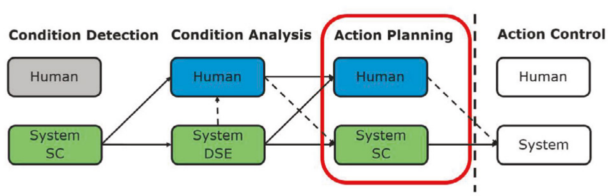

Fig. 1. Self-controlling path-planning system (DNV GL, 2018).

task, the following requirements have to be considered as stated in the guideline (DNV GL, 2018): "Based on the object classification information, the system has capabilities to calculate an updated passage plan in accordance with COLREGs that are equivalent or better than that of a navigator on board the vessel". In other words, the path-planning system must be capable of adapting to the environment, and hence generating a suitable path in compliance with COLREGs. Additionally, the remote operator needs to be provided with sufficient information to be able to derive independent conclusion on the optimal action. For instance, the collision avoidance system should clearly indicate the updated plan before a control action, giving the remote operator enough time to make his/her own analysis and intervene if necessary. If the navigation task becomes excessively complex for the system to handle, the vessel or the operator should have the option to bring the vessel to a Minimum Risk Condition (MRC). MRC is defined as a state that causes least risk to life, environment and property, and a state the vessel should enter when an abnormal situation occurs. The system requirement for an autonomous path-planning can therefore be summarized to the following two conditions:

(1) The system is expected to comply with COLREGs.
(2) The system is expected to offer transparency of the planned maneuvering.

2.2 COLREGs

To prevent collision in ocean traffic, IMO established the convention International Regulations for Preventing Collision at Sea, which defines a set of navigation rules to be followed by the vessels and crews sailing the sea (Lloyd's Register, 2005). Therefore, in order to achieve an adequate path-planning system, the legal rules have to be incorporated in the model. In total COLREGs covers 40 rules and regulations related to different scenarios at the sea, as well as requirements on equipment to prevent collision. In this paper we reduce the problem to only consider Rule 14 related to head-on situations. Rule 14a) states that the vessel shall alter the course starboard to avoid collision during head-on situation (see fig. 2 for illustration). Additionally, if there is any doubt whether such a situation exist, the vessel shall assume that it does exist and act accordingly, as emphasized by rule 14c).

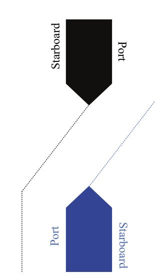

Fig. 2. Head-on situation as defined by COLREGs rule 14.

2.3 Vessel model

To describe the vessel motion in 6 degree-of-freedom a mathematical process model as defined in Fossen (2011) can be applied. The model incorporates both the vessel dynamics and kinematics of a real vessel and is given as:

$$M\dot{\nu} + C(\nu)\nu + D(\nu_r)\nu_r + G\eta = \tau \quad (1)$$

$$\dot{\eta} = R(\psi)\nu \quad (2)$$

where $\eta = [x,\, y,\, z,\, \phi,\, \theta,\, \psi]^T$ represents the position and heading in the Earth fixed coordinates, and $\nu = [u,\, v,\, w,\, p,\, q,\, r]^T$ is the generalized velocity in the body-fixed frame. Since the environmental forces are neglected τ only considers the control forces, and hence simplifies to $\tau = [f_u,\, f_v,\, f_r]^T$ representing the surge- and sway-forces and the yaw-moment produced by the controller. Further, matrix M represents the inertia and added-mass. The C matrix denotes the Coriolis and centripetal matrices for rigid-body and added-mass. In general, the damping matrix D of a vessel can be divided into a non-linear component D_{NL} and a linear component D_L. For increasing speed and turbulent flow, the linear damping can be considered distinguishable compared to the contribution from the non-linear damping. Correspondingly, for velocities close to zero, the linear damping becomes more dominant than the non-linear component. The generalized restoring matrix G consist of the linear gravitation and buoyancy force coefficients. Finally, the $R(\psi)$ denotes the rotation matrix. Alternately, the vessel model can be further simplified to apply in 3 degrees-of-freedom resulting in a control plant model. Unlike the process plant model which can become fairly complex, the control plant distinguishes the non-linear components, and hence often used during development of controllers. Note that despite being a simplification, the control plant model still incorporates the essential behavior of a vessel.

2.4 Path-following

In general, the main goal of any arbitrary vessel is to get from an initial location to a desired location. This is often accomplished based on one of two maneuvering strategies:

- **Tracking**: In tracking the vessel traces a target or moving point through a trajectory to get to the desired destination. The trajectory describes the motion of the vessel derived mathematically as a geometric path or the position as a function of time.
- **Path-following**: In path-following the vessel aims to converge to and follow a predefined path, which is independent of time.

However, in Skjetne (2005) an alternative method named the Maneuvering problem was proposed. In the Maneuvering problem the path-following-task is divided into two sub-tasks defined as the geometric task and the dynamic task. In the geometric task a desired path y_d is defined for the vessel to follow. The dynamic task on other hand, introduces constraints on the dynamic behavior of the vessel while following the path, in particular on the cruising speed or acceleration. Hence, the aim of the latter is to avoid any undesirable dynamic behavior of the vessel during path-following. Note that the tracking problem utilize the characteristics of both geometric- and dynamic constraints. But unlike the tracking problem, the sub-tasks are not equally weighed in the Maneuvering problem. Consequently, when the vessel faces difficulties while following the path, the dynamic task is sacrificed to improve the path-following. In general, for a system with output y $\in R^m$, we can define the points in the desired path as the set:

$$\mathcal{P} : \{y \in R^m : \exists s \in R \ s.t. \ y = y_d(s)\} \quad (3)$$

where $y_d(s)$ represents the desired path parametrized by the continuous path variable s. We can now mathematically formulate the two sub-tasks as following:

(1) **Geometric task:** For any continuous function $s(t)$, force the output y to converge to the desired path $y_d(s)$:

$$\lim_{t\to\infty} |y(t) - y_d(s(t))| = 0 \quad (4)$$

(2) **Dynamic task:** Satisfy one or more of the following assignments:

(a) **Time assignment:** Force the path variable s to converge to a desired time signal $v_t(t)$:

$$\lim_{t\to\infty} |s(t) - v_t(t)| = 0 \quad (5)$$

(b) **Speed assignment:** Force the path speed $\dot{s}$ to converge to a desired speed $v_s(s(t), t)$

$$\lim_{t\to\infty} |\dot{s}(t) - v_s(s(t), t)| = 0 \quad (6)$$

(c) **Acceleration assignment:** Force the path acceleration $\ddot{s}(t)$ to converge to a desired acceleration $v_a(\dot{s}(t), s(t), t)$

$$\lim_{x\to\infty} |\ddot{s}(t) - v_a(\dot{s}(t), s(t), t)| = 0 \quad (7)$$

2.5 Path parametrization

For the geometric task, the desired path y_d is often represented either as a straight-line path or a curved path. The former, also known as way-point tracking, is usually preferred due to its simplicity. However, in situations where changes in the heading are considerably large, discontinuity may be introduced, as discussed in Fossen (2011). This can be avoided by applying interpolated paths, such as curved paths. For curved path-following the entire desired path is defined by a geometric curve parametrized by a continuous path variable s. There are numerous ways to design such a curve, but in this paper, we consider the Bézier curve. A Bézier curve is a parametric curve based on the Bernstein polynomials and mainly used in Computer Aided Geometric Design (Sederberg, 2012). It was originally introduced by Dr. Pierre Bézier during the 1960s for sketching the design of Renault cars. In general, a Bézier curve of n degree consist of $n + 1$ control points P_0, P_1,..., P_n as observed in fig. 3. The Bézier curve is designed such that it always passes

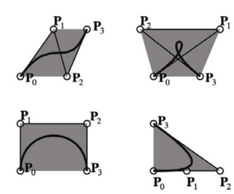

Fig. 3. Examples of Bézier curves and the respective control polygons (Sederberg, 2012).

through the first- and last points. It has the additional property of being tangential to the control polygon at the endpoint. The control polygon is the shaded polygon created by connecting the control points in ascending order. Furthermore, the curve can never be outside the control polygon. The points between the endpoints help to shape the curvature of the path, and does not necessarily lay on the actual curve.

2.6 Supervised learning

Despite the fact that Supervised learning have gained huge popularity the last decade, the approach is not considerable new. The most elementary Supervised learning model is the Linear Regression well-known from fundamental math courses. In general, a Linear Regression model aims to describe the relation between a set of explanatory variables, also known as *features*, and an observation or *target variable*. For instance, assume we want to find or *predict* an observation Y, and the only available information is the p

number of features x_i. Then Linear Regression states that the following mathematical relationship exist:

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \epsilon \tag{8}$$

where ϵ represents a random error term independent of x with zero mean. The solution of the problem reduces to estimating the coefficients β_0, β_1, ..., β_p. This step is often termed as *fitting* or *training* of the model, and usually accomplished by applying an optimization algorithm such as least squares. Today there exists a large group of Supervised models, in particular Support Vector Machine, K-mean, Decision Trees, Lasso Regression and so on (James et al., 2013). Although they share the same principle, they are considerably more accurate and efficient algorithms compared to Linear Regression. The general model development of Supervised learning can be summarized into the following steps:

(1) **Data collection and -preparation**: As a consequence of computer- and information technology advancement, massive quantities of data is available today. However, in most cases the data is stored in a useless state, and only a small amount of the data is usually considered valuable. Thus, data preparation is an essential step involving cleansing, manipulation and assembling of the collected data into an applicable state. This step is also termed as *feature engineering*, and often considered the most important part of the process, as selecting the correct data decides the final performance of the model.
(2) **Model selection**: In 1997 David Wolpert and William Macready stated the *No free lunch-theorem* (Ciuffo and Punzo, 2014). The theorem states that there does not exist a particular model or algorithm that is applicable for all problems. In other words, a model that performs well in one problem, may be unsuitable in a different domain. Hence, in model selection one usually has to evaluate and compare different models to uncover the most suitable model. In general, one selects a set of models that are trained using the data extracted from the previous step. Each model has its own set of hyperparameters that are optimized during training. Typically, a common criterion is defined across the models such as minimizing the mean-squared error (MSE). Model selection often involves several iterations of model tuning and -evaluation before one is finally selected.

3. IMPLEMENTATION

The simulation of a vessel is carried out using a Simulink-model based on the process plant model of the physical research vessel R/V Gunnerus, developed at Norwegian University of Science and Technology (NTNU) by the Department of Marine Technology. The vessel model is used to simulate both the own vessel and target vessel. The digital twin on other hand, is implemented based on the control plant model of R/V Gunnerus. Since the advanced machine learning libraries are mainly developed in Python, the simulation models of the vessels are converted to a Functional Mock-up Unit-format using the open-source library FMI Kit (Catia-systems, 2019a). Consequently, the Simulink-models can be integrated with the path-planner in the Python environment using FMPY (Catia-systems, 2019b). In similar fashion, to steer the vessel from A to B, a controller based on the Maneuvering problem is implemented in Simulink. Notice that the controller can be shown to be UGES using backstepping as demonstrated in Vasanthan (2020). Hence, the vessel is guaranteed to converge to the desired path. The speed assignment is selected as the most suitable dynamic task, where the reference speed is set to 5 m/s. Further, to solve the geometric task a parametrization based on cubic Bézier curve is implemented. The explicit form of the cubic curve can be written mathematically as:

$$\begin{aligned} \mathbf{B}(s) = (1-s)^3\mathbf{P}_0 + 3(1-s)^2 t\mathbf{P}_1 \\ +3(1-s)t^2\mathbf{P}_2 + s^3\mathbf{P}_3, \\ 0 \leq s \leq 1. \end{aligned} \tag{9}$$

Since the initial position and the end-destination of a vessel is normally known, it is assumed that the endpoints of the curves are pre-defined. Therefore, the remaining task is to determine the two control points between the endpoints. As discussed previously, the aim is to choose a path that satisfies the criteria defined by DNV GL and COLREGs rule 14. Hence, the control points have to be picked such that the following constraints are fulfilled:

- Take the shortest possible path.
- Avoid collision at any cost.
- Comply with seafaring rules, more specifically rule 14 in COLREGs.
- If the resulting course would result in collision, initiate re-planning.

To accomplish this, a score paradigm is introduced with respect to the constraints. The idea is to give each selection of control points a score based on how well the generated path complies with the constraints. The resulting relationship is used to select a suitable path based on the information of the vessel and any present target vessel. To achieve this, we specify the control points along with the vessel states as the features, while the score is chosen as the target variable. Note that the own vessel states are defined relative to the goal position and the target vessel, respectively. In general, features with larger magnitude have stronger impact on the resulting prediction. Hence, each feature has to be normalized individually.

To address the first constraint a circular safe-zone is established enclosing the target vessel with a predefined radius. The radius is chosen such that the own vessel can avoid any doubtful situation as defined in COLREGs rule 14c), devoting time to take early actions. If the own vessel crosses the safe-zone, a strict negative penalty $R_{Collision}$ is given. To satisfy the second condition, a reward-zone is generated in compliance with COLREGs 14a). That is, to force the own vessel to maneuver starboard during a head-on situation. Hence, the reward-zone is always established on the starboard-side relative to the velocity vector-direction of the target vessel as illustrated by the enclosed red rectangle in fig. 4. When the own vessel enters the reward-zone, it begins to accumulate a small reward $R_{COLREGs}$. Note that since the endpoints of the curves are pre-defined and the reference speed is constant, the autonomous path-planner cannot generate a path that remains inside the zone. To meet the third constraint, we note that the shortest path from A to B is a straight-

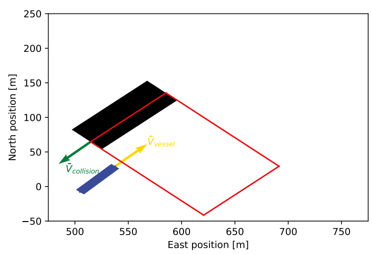

Fig. 4. The blue and black rectangles represent the own- and target vessel, respectively. The enclosed red rectangle illustrates the reward zone in terms of COLREGs 14a), and hence placed on the starboard-side relative to the own vessel.

line. Hence, whenever a target vessel is not nearby, the ideal selection of control points should result in a straight-line. However, when a target vessel is present, it is still desired that the own vessel maneuvers as close to the straight-line path as possible to minimize the distance travelled. Therefore, a Gaussian reward-function R_{Path} is introduced, as defined in Bell and Lekkas (2018). In general, the cross-track represents the error between the own vessel and the desired straight-line path, and is given by the equation:

$$y_e = -sin(\psi_d)(x(t) - x_0) + cos(\psi_d)(y(t) - y_0) \quad (10)$$

$$\psi_d = atan2(y_1, x_1) \quad (11)$$

where ψ_d is the heading of the path, (x(t), y(t)) is the position of the vessel, $(x_0,\ y_0)$ and $(y_1,\ x_1)$ are the initial and final points of the desired straight-line path, respectively. Hence, minimizing y_e is equivalent to the ship converging to the shortest path. The resulting Gaussian reward function then becomes:

$$R_{Path} = ae^{-\frac{y_e^2}{2\sigma}} \quad (12)$$

y_e is the computed cross-track error, σ is the standard deviation of the cross-track error and a is the maximum attainable reward, where latter has to be manually chosen. The aim is to choose the reward such that the own vessel stays close to the straight path. The Gaussian reward function represents a Gaussian curve with amplitude a, and a standard deviation σ as seen in fig. 5. Studying

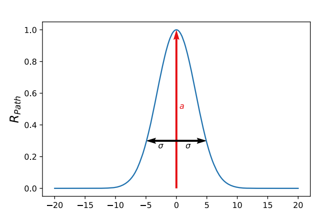

Fig. 5. Distribution of R_{Path} over y_e for σ=5 and a=1.

the figure, it is evident that the model only gains a considerable reward when the cross-track error is below 5 [m]. Bear in mind that each respective reward function has to be tuned relative to the others. For example, picking a large reward $R_{COLREGs}$ will stimulate the path-planner to generate a path that maneuvers next to the obstacle, such that the accumulated reward is increased, even if the approaching vessel maintains a safe distance. Similarly, choosing a small penalty $R_{Collision}$ relative to R_{Path}, may encourage the path-planner to generate a straight path that results in collision. This leads to the following relationship:

$$R_{Collision} = \begin{cases} -100 \text{ if inside safe-zone,} \\ 0 \qquad \text{otherwise.} \end{cases} \quad (13)$$

$$(14)$$

$$R_{COLREGs} = \begin{cases} 20 \text{ if inside reward-zone,} \\ 0 \quad \text{otherwise.} \end{cases} \quad (15)$$

$$(16)$$

$$R_{Total} = R_{Collision} + R_{COLREGs} + R_{Path} \quad (17)$$

To generate training data for the training process, the digital twin is utilized. Initially, an indiscriminate set of control points are chosen. After each simulation, the selected control points, and the states of the vessel along with the resulting score are retained as training data. For model selection 400 000 training samples were generated. A wide selection of models was compared using cross-validation and standard hyperparameters. Fig. 6 shows how the MSE on the test set decreases as the training sample increases. Eventually, the Gradient Boosting Regressor was chosen as the most adequate algorithm based on hyperparameter-tuning. The resulting model is then

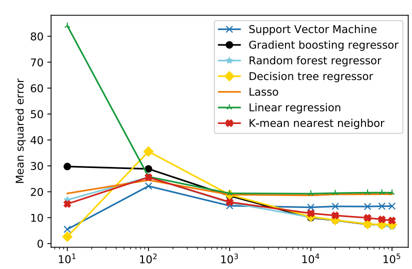

Fig. 6. Benchmarking a set of regression models in terms of MSE.

applied on the own vessel. In general, to generate a path from A to B, an indiscriminate selection of control points C_1 and C_2 are proposed to the model, scaled to the environment. Based on the vessel- and obstacle states, the model predicts a score on how well the resulting path will satisfy the constraints. This process is repeated iteratively until the model returns an acceptable score. Hence, an optimal number of iterations K has to be found, such that the constraints are always satisfied. In general, larger number of iterations will result in more optimal control points, but in exchange for run-time. To find the optimal number, K is initially set to a large number. Then a path-planning problem is presented to the model, and the iteration number returning the largest score is noted. This process was repeated 10 000 times and is presented in fig 7. It can be observed that at most 8 000 iterations were required to find the best selection of control points for the corresponding problem. The mean and standard deviation was found to be 1 565.12 and 1 249.10, respectively. Hence, K is set to minimum 3 000 to cover the significant areas, and if not, until a positive score is returned. Despite being

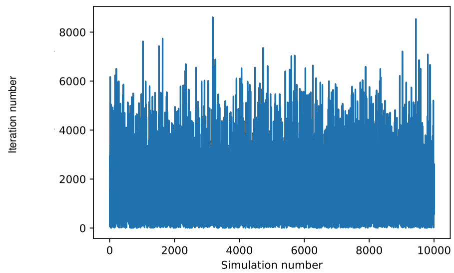

Fig. 7. Number of iterations before optimal control points were found for 10 000 simulation cases.

a large number, in practice the process equals to less than ten seconds on an average computer.

Finally, to accomplish the ability to re-plan, the digital twin is implemented inside the path-planner. For a given time-frequency, the digital twin will simulate the resulting motion based on the current states of the own- and target vessel combined with the generated path. The model is then capable of forecasting a potential collision. If that is the case, the digital twin will prompt the model to find a new set of control points and generate a new path for the current environment variables.

4. SIMULATION RESULTS AND DISCUSSION

First scenario In general, the initial condition of the own vessel was set to $(x_0,\ y_0,\ \psi_0) = (0,\ 0,\ 0)$. To verify that the vessel was capable of choosing the shortest path, the starting point and endpoint of the path were set to $(x_0, y_0) = (0, 0)$ and $(x_1, y_1) = (3200, -200)$, respectively. The generated path and resulting vessel maneuvering can be studied in fig. 8 and 9.

Second scenario In the second scenario a target vessel with real vessel dynamics was introduced. The heading of the target vessel was set specifically to interfere with the own vessel. The initial point of the path was defined as $(x_0,\ y_0) = (0,\ 0)$ with endpoint $(x_1,\ y_1) = (2000,\ 300)$. As studied in fig. 10 and 11, the generated path makes the own vessel correctly maneuver on the starboard-side of the target vessel.

Third scenario The third scenario was similar to previous scenario, but the target vessel was defined such that it was unlikely to collide with the own vessel. Hence, the expected behavior was a close to straight-line path as successfully showcased in fig. 12 and 13. The initial point of the path was set as $(x_0,\ y_0) = (0,\ 0)$ and endpoint $(x_1, y_1) = (3400,\ 300)$.

Fourth scenario To evaluate the performance of re-planning, an unexpected change in the heading of the target vessel was introduced midway into the simulation. The initial point and endpoint of the path was set as $(x_0, y_0) = (0,\ 0)$ and $(x_1,\ y_1) = (2800,\ 0)$, respectively. The blue marks in fig. 14 represents the spatial instants when the digital twin recognized probability of a collision, and hence initiated the re-planning of a modified path. Notice the discontinuity in the trajectory of the own vessel, which is naturally introduced due to the sudden change in the heading of the target vessel.

5. CONCLUSION

In this paper a novel solution for the autonomous path-planning problem was proposed. The aim of the solu-

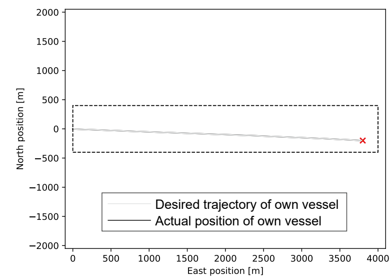

Fig. 8. First scenario: Straight-line path-planning.

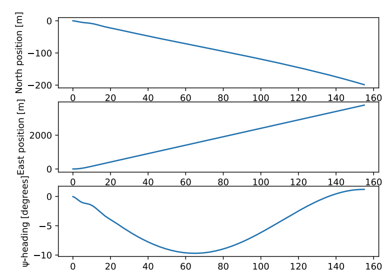

Fig. 9. First scenario: Plot of North-, East- and ψ-position.

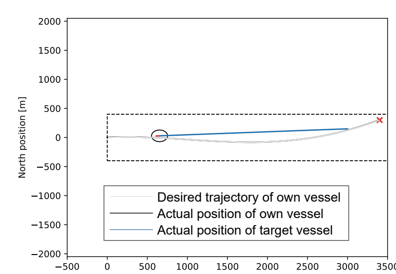

Fig. 10. Second scenario: Obstacle-avoidance.

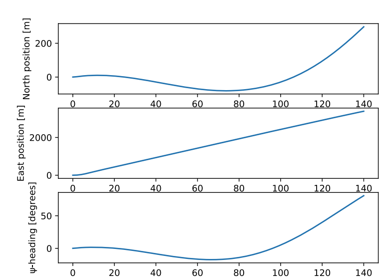

Fig. 11. Second scenario: Plot of North-, East- and ψ-position.

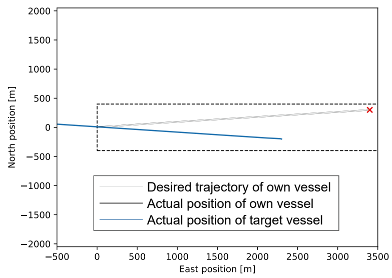

Fig. 12. Third scenario: Obstacle-avoidance. Note that the target vessel is outside the environment by the time the own vessel reaches destination.

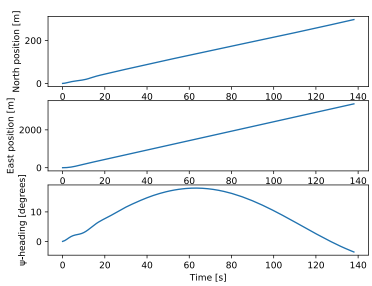

Fig. 13. Third scenario: Plot of North-, East- and ψ-position.

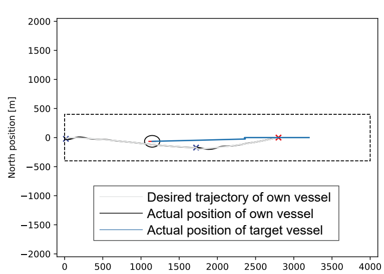

Fig. 14. Fourth scenario: Obstacle-avoidance with re-planning.

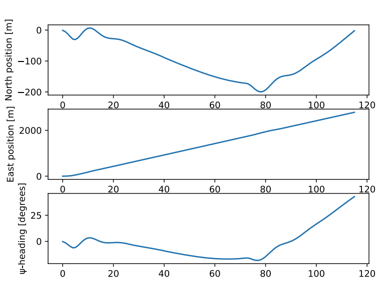

Fig. 15. Fourth scenario: Plot of North-, East- and ψ-position.

tion was to demonstrate the ability to generate satisfying paths considering collision-avoidance while removing any uncertainty, in particular introduced by a Deep learning-based approach. As a consequence, an autonomous path-planner based on Supervised learning was developed. The constraints of the model were established based on the criteria proposed by DNV GL for an autonomous path-planner in combination with COLREGs rule 14. However, a noteworthy drawback of Supervised learning is the necessity for exceedingly large amount of training data. Hence, a digital twin of the own vessel was developed based on a control plant model. In addition, to detect a potential collision, and thereby initiating the re-planning step, the digital twin was implemented in the path-planner. The simulation was carried out using a Simulink-model of a real physical vessel, R/V Gunnerus. The selected simulation cases demonstrated promising results in satisfying the constraints. In terms of further work implementation of environment forces and application of additional COLREGs rules have to be considered.

ACKNOWLEDGEMENTS

This work was supported by the Research Council of Norway through the Centres of Excellence funding scheme, project no. 223254 - NTNU AMOS.

REFERENCES

Aronsen, M.G. (2019). Path planning and obstacle avoidance for marine vessels using the deep deterministic policy gradient method. Master thesis.

Bell, A.M. and Lekkas, A.M. (2018). Curved path following with deep reinforcement learning: Results from three vessel models. *Conference: OCEANS 2018 MTS/IEEE Charleston*.

Blindheim, S., Gros, S., and Johansen, T.A. (2020). Curved path following with deep reinforcement learning: Results from three vessel models. *IFAC-PapersOnLine*, 53.

Buhrmester, V., Muench, D., and Arens, M. (2019). Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey.

Catia-systems (2019a). FMIKit-Simulink. URL `https://github.com/CATIA-Systems/FMIKit-Simulink`.

Catia-systems (2019b). FMPy. URL `https://github.com/fmi-tools/FMPy`.

Ciuffo, B. and Punzo, V. (2014). "No Free Lunch" Theorems Applied to the Calibration of Traffic Simulation Models. *Intelligent Transportation Systems, IEEE Transactions on*, 15, 553–562.

DNV GL (2018). Class guideline: Autonomous and Remotely Operated Ships. URL `http://rules.dnvgl.com/docs/pdf/dnvgl/cg/2018-09/dnvgl-cg-0264.pdf`. DNVGL-CG-0264 Edition September 2018.

Dulac-Arnold, G., Mankowitz, D., and Hester, T. (2019). Challenges of Real-World Reinforcement Learning. *Modeling, Identification and Control*, 30.

Eriksen, B.O. and Breivik, M. (2017). Mpc-based mid-level collision avoidance for asvs using nonlinear programming. *Conference: IEEE Conference on Control Technology and Applications*.

Fossen, T.I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley Sons.

Ghassemi, M., Naumann, T., Schulam, P., Beam, A., Chen, I., and Ranganath, R. (2020). A Review of Challenges and Opportunities in Machine Learning for Health. *AMIA Joint Summits on Translational Science proceedings. AMIA Joint Summits on Translational Science*, 2020, 191–200.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R.* Springer.

Kelly, C.J., Karthikesalingam, A., Suleyman, M., Corrado, G., and King, D. (2019). Key challenges for delivering clinical impact with Artificial Intelligence. *BMC Medicine*, 17(1), 195. URL https://doi.org/10.1186/s12916-019-1426-2.

Kongsberg Group (2020). Autonomous ship project, key facts about Yara Birkeland. URL https://www.kongsberg.com/maritime/support/themes/autonomous-ship-project-key-facts-about-yara-birkeland/?OpenDocument=.

Li, F. and Du, Y. (2018). From AlphaGo to Power System AI: What Engineers can learn from solving the most complex problems Board Game. *IEEE Power and Energy Magazine*, 16, 76–84.

Lloyd's Register (2005). COLREGS - International Regulations for Preventing Collisions at Sea. URL http://www.mar.ist.utl.pt/mventura/Projecto-Navios-I/IMO-Conventions%20(copies)/COLREG-1972.pdf.

Lloyd's Register (2017). LR Code for Unmanned Marine Systems. URL https://www.lr.org/en/unmanned-code/.

Luman, Z., Roh, M.I., and Lee, S.J. (2019). Control method for path following and collision avoidance of autonomous ship based on deep reinforcement learning. *Journal of Marine Science and Technology*.

Mendi, A., Erol, T., and Dogan, D. (2021). Digital Twin in the Military Field. *IEEE Internet Computing*, PP, 1–1.

Sederberg, T.W. (2012). *Computer Aided Geometric Design*. Brigham Youth University, All faculty publication.

Singh, Y., Sharma, S., Sutton, R., and Hatton, D. (2017). Optimal path planning of an unmanned surface vehicle in a real- time marine environment using a dijkstra algorithm. *Conference: The International Conference on Marine Navigation and Safety of Sea Transportation (TRANSNAV 2017)*, 399–402.

Skjetne, R. (2005). The Maneuvering Problem. Phd thesis.

Stangvik, E.O., Skjetne, O.L., Byermoen, T., Alsaker-Nøstdahl, E., and Vikøyr, H. (2019). Krigsskipet som krasjet og sank. URL https://www.vg.no/spesial/2018/helge-ingstad-ulykken/.

Sørensen, A.J. (2018). Marine Cybernetics Towards Autonomous Marine Operations and Systems. Lecture notes.

Vallestad, I.J. (2019). Path Following and Collision Avoidance for Marine Vessels with Deep Reinforcement Learning. Master thesis.

Vasanthan, C. (2020). Combining Supervised Learning and Digital Twin for Path-planning with Dynamic Obstacle-avoidance. Master thesis.

Zaccone, R. (2021). Colreg-compliant optimal path planning for real-time guidance and control of autonomous ships. *Journal of Marine Science and Engineering*, 9, 405.

Zheng, H., Negenborn, R.R., and Lodewijks, G. (2014). Trajectory tracking of autonomous vessels using model predictive control. *IFAC Proceedings Volumes*, 47(3), 8812–8818. 19th IFAC World Congress.