Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-16 (2021) 57–62

Path Following Control for Autonomous Ship using Model Predictive Control

Victor C. Codesseira *, **Eduardo A. Tannuri** **

* *Escola Politécnica, University of São Paulo, Brazil (e-mail: victor.codesseira@gmail.com)*

** *Escola Politécnica, University of São Paulo, Brazil (e-mail: eduat@usp.br)*

Abstract: Surface ships are, generally, underactuated, as they have more degrees of freedom than actuators, which lead to several difficulties in controlling them. Model Predictive Control (MPC) is a modern control technique, which seeks to solve some of the problems that arise with classical control techniques by optimizing inputs over a finite period of time. The central goal in this study was implementing a MPC controller for path following, to be tested in a simulator developed by the Numerical Offshore Tank (TPN) of the University of São Paulo. With such controller, we achieved a robust and efficient control, capable of adapting to different vessels and environmental conditions.

Keywords: Model Predictive Control; Vessel; Ship; Modern Control; Path Following.

1. INTRODUCTION

Classical control techniques, although used, are not the most adequate for ship track control. That is the case because they are underactuated systems, which have 3 degrees of freedom (if we consider planar movement), and usually a single control variable, with at most 2 (rudder and engine). Therefore, more advanced control techniques are needed.

A possible technique we can use is Model Predictive Control (MPC), which optimizes the inputs of the system, taking into consideration the dynamic model, as well as its response to control efforts. This technique also allows us to take into consideration restrictions in the actuators, such as a maximum rudder angle, and the controller will never decide for a higher actuation effort than is possible, which would lead to actuator saturation.

The flexibility of MPC regarding actuator restrictions and the ability to optimize the cost function for different situations are some of the reasons why it is widely used in autonomous systems such as cars and ships. For some time, the computational cost of the operations used in MPC made it hard to use as a real-time control system. But the recent increase in processing power and decrease in costs have made it so that MPC is now a viable strategy, that doesn't need a large computer as a controller.

The goal of this project is to develop a path following controller for an autonomous ship, capable of following a path set by waypoints, using MPC, and test its viability using a Python-language ship maneuvering simulator of TPN-USP laboratory, named PyDyna. We seek to keep a ship inside a path determined by buoys, using the controller to calculate the values for the ship rudder, and test in the simulator for a real case, for navigation in a restricted channel.

2. LITERATURE REVIEW

2.1 Model Predictive Control

As indicated by the name, Model Predictive Control is a technique of control that takes into account the dynamic model of the system we wish to control, over a time horizon, to decide a control action. MPC started in industry, was developed by industry, and applied in industry, as said by Annamalai (2012), which contributed to its success.

One of the main advantages of MPC is the ability to take into account different types of control limits. Other control approaches may take into account the limitations in the states of the system, but with MPC we can also take into account inherent limitations of the actuators, such as saturation and maximum actuation speed, and we can also consider uncontrolled dynamics in the system, such as a ship's roll. For instance, Siramdasu and Fahimi (2013) talks about an experiment in which a sliding control was used, and the system suffered from instability problems when initial conditions where too far apart from the goal state, so a PID controller was necessary on system startup. With MPC, that problem doesn't arise, because at the moment the controller decides the optimal actuation effort, it takes into consideration the actuators limits.

MPC works, basically, simulating a sequence of actuation efforts, over a time period called sliding horizon (because the time period advances, alongside the simulation). Then, using a numeric solver, a cost function is minimized over the entire horizon, which defines an optimal control sequence. Only the first value from this sequence is applied on the system, and then the horizon slides over to the next step, and we repeat the whole operation again, as shown in Zheng et al. (2014).

10.1016/j.ifacol.2021.10.073

The cost function defines which type of control is to be done. Usually, for path following, the cost function is the tracking error over the time horizon, as in Oh and Sun (2010); Zeng et al. (2019), but it can be, for instance, a function that minimizes the energy used for actuation, in embedded systems with limited battery supply.

The minimization of this cost function is the most computationally heavy operation in the control loop, and must happen every step of the program, so it is an interesting point of optimization of the controller, and Liu et al. (2020) shows the use of neural networks to speed up the minimization.

2.2 Dynamic Model

In any MPC application, the dynamic model of the system to be controlled is the most important part. If we use the wrong model, the controller will not be able of actuating effectively, and in worse cases, could even take the system to instability. Therefore, one of the most important parts in the implementation of MPC is the theoretic modeling of the controlled system, and the identification of its parameters.

For a ship, we usually adopt the dynamic model of a underactuated system, with 3 degrees of freedom and only 1 or 2 actuators, one of the reasons of the control being so difficult. Some models with higher complexity, as in Zhang et al. (2017); Li et al. (2009), take into account more degrees of freedom, such as roll, which makes it even harder to develop a controller. A possible modeling approach is given in Skjetne et al. (2004), as well as the parameters necessary for simulating CyberShip II. This model is commonly used as a benchmark for control systems.

A possible approach for approximating the model of a dynamic system is the implementation of an Extended Kalman Filter (EKF), which besides approximating the states of the system, also approximates the model parameters. Such an approach can be seen in Perera et al. (2015), in which the EKF is used to approximate the parameters of a ship, using a second order Nomoto model.

2.3 Line of Sight

MPC, like any other control, seeks to bring a system to a desired state, or a goal. In the case of a ship, the most probable controls to implement are trajectory tracking or path following. In path following, the controller receives a set of points that define the desired motion. In trajectory tracking, the controller also receives times for the points, so it must follow a path, while also respecting timing constraints, as defined in Karimi and Lu (2021).

Usually the path is given by a sequence of points, or waypoints, which define the path to be taken by the ship. In the simplest case, the system will adopt a course between the current position of the ship and the current waypoint, and upon arriving in this waypoint, will change its focus to the next one. However, that is not an optimal trajectory, in speed or energy. A possible upgrade is to define a radius around the current waypoint. As soon as the ship enters this radius, the goal is switched to the next waypoint, as show in Fossen (2011).

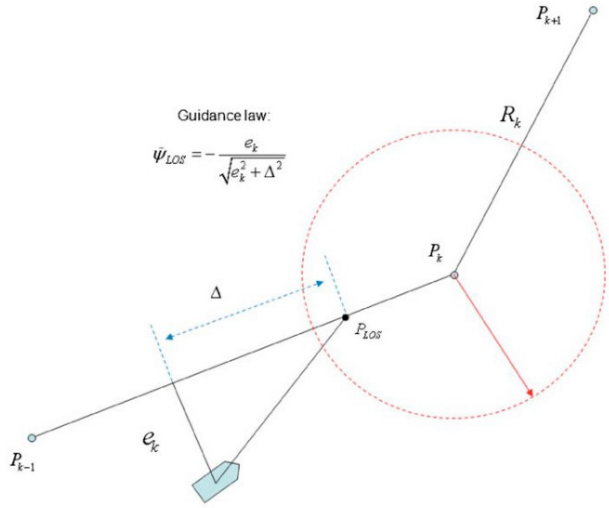

Fig. 1. LOS example - Source: Oh and Sun (2010)

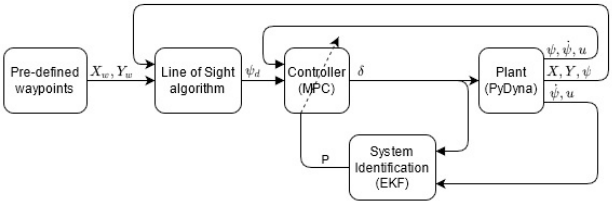

Fig. 2. Block Diagram for the Controlled System

A more refined approach, that is often used, is called Line of Sight (LOS). In this method, a line is created between the last and the current goals of the ship. Then, the controller finds the closest point in this line, from the ship, and defines the course to a position ahead, by a distance called lookahead. This way, the controller is able to follow the path in a smoother and more efficient way. This procedure is illustrated in Figure 1. An alternative, that is used to keep course even with constant perturbations (sea currents, for instance), is the adoption of an integral action in the angle given by LOS. This approach is called ILOS, and is given in Zeng et al. (2019).

3. METHODOLOGY

In figure 2, we can see the block diagram for the controlled system. The waypoints the ship will follow are pre-defined, and will feed the LOS algorithm with planar coordinates for each waypoint, X_w and Y_w. This algorithm then uses those coordinates, plus the current position of the vessel, to obtain a desired angle, ψ_d. With that angle, the vessel's current heading, rotational and linear velocities (ψ, $\dot{\psi}$ and u, respectively), the controller, using MPC, can obtain the best value for the rudder angle, δ.

The plant then reacts to that input, and the updated velocities, along with the input (δ), are used to obtain the most up to date parameters (P) for the simplified model used in the controller. In that way, we have a controller that is capable of adapting to different conditions online, by varying the parameters for the model, depending on how the plant reacts to the input.

Fig. 3. Parameters obtained using the filter

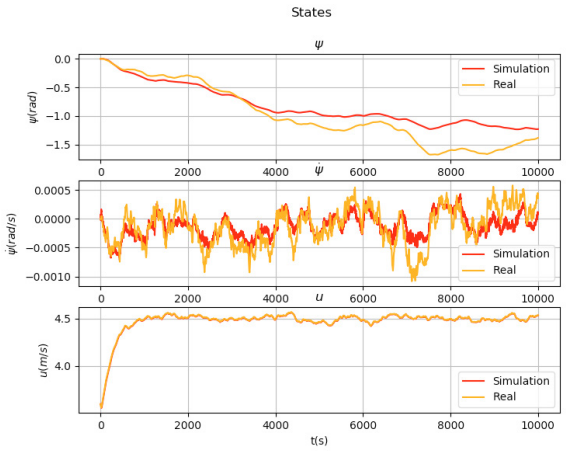

Fig. 4. States with PyDyna ("Real"), and with parallel simulation, pre-initialized ("Simulation")

The model used in the controller is discussed further in section 4.1. For testing purposes, the plant used will be the PyDyna simulator. The PyDyna simulator is a ship maneuvering simulator implemented on python based on the mathematical model adopted in the TPN-USP Ship Maneuvering Simulation Center, as show in Tannuri et al. (2014).

4. IMPLEMENTATION

4.1 Obtaining the dynamic model

The first step in developing a MPC controller is obtaining a dynamic model that represents the system. For that, we used an Extended Kalman Filter, as shown in Perera et al. (2015). Using this approach, we can assume a certain dynamic for our controlled system, and use the EKF to obtain the value of the constants. Several dynamic models where tested, but in the end we adopted the well-known First Order Nomoto Model given by

$$\ddot{\psi} = a\dot{\psi} + b\delta \tag{1}$$

in which ψ is the current heading angle of the ship, δ is the rudder angle are constants not yet defined. The ship's longitudinal dynamic follows the simplified dynamics

$$\dot{u} = cu + dr^2 \tag{2}$$

u being the linear speed of the ship in the main axis, r the rotation of the engine and c and d unknown constants. Applying the methodology defined in Perera et al. (2015), we were able to achieve satisfactory results for system identification. In figure 3 we are able to see the evolution of the 4 parameters of the controller over the simulation.

In figure 4, we can see the results of a simulation using the dynamic model obtained by the Kalman filter, running in parallel with PyDyna's simulation. It is worth noting that, in this simulation, we initialized the parameters in the Kalman Filter with the ones obtained previously. That way, we can see that, mainly in the first 200 seconds (which are more than we need for MPC), we have a good approximation for the dynamic of the ship using the Kalman Filter.

4.2 Implementing the controller

Having the dynamic model, we could implement the MPC controller. For that, we used the function *minimize* of the library *scipy*. That function allows us to find the input values that minimize the output of a certain function. For MPC specifically, the function to be minimized is the cost function, that must take into account those inputs and the model of the system to calculate some cost.

In order to get the best control inputs for the system, we first assume that the last input applied will continue being applied over the entire span of the control horizon. Then, we simulate the evolution of the system, over the horizon, considering those inputs and the model given above, in section4.1, using an ODE integrator. Using the simulated states, we can calculate the cost function, which can be written as

$$C = \sum_{i=1}^{H} w_1 e_{\psi i}^2 + w_2 \dot{\psi}_i^2 + w_3 u_i^2 + w_4 \dot{u}_i^2 + w_5 r_i^2 + w_6 \dot{r}_i^2 \tag{3}$$

Where H is the amount of steps on the time horizon, e_ψ is the angular heading error, and w_{1-6} are calibrated weights. So overall, the cost for this specific sequence of inputs is a weighted sum between the values of the tracking error, the derivative of that error, the amplitude and derivatives of the control inputs. With that, we are not only optimizing the system to converged to the desired course, but also to keep the value of the derivative low, to avoid problems with instability and overshoot. We are also minimizing the values of the control inputs, to avoid sudden variations and saturation.

The *minimize* function will then repeat these same operations for slightly different inputs, nudging the input values for the simulations in different ways, but never going over the physical limits of the actuators, until it reaches a local minimum. After that, we take the optimal sequence of inputs, and apply the first to our plant.

After implementation, the controller was then tested with a sequence of waypoints, that using LOS, give the desired heading for the ship at any given moment. We can see in figure 5, the controller is able to follow the path given by the waypoints, with no significant deviations.

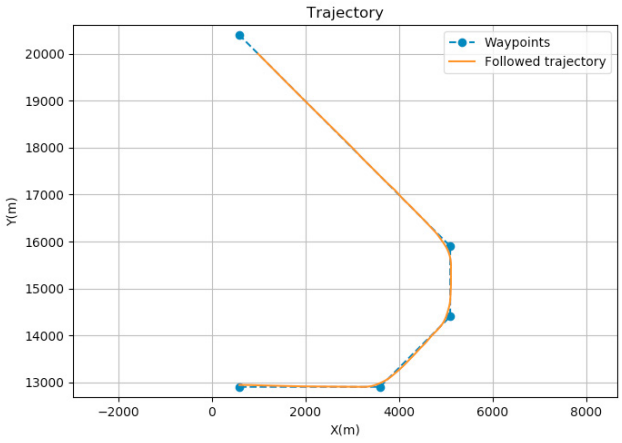

Fig. 5. Path followed in test case

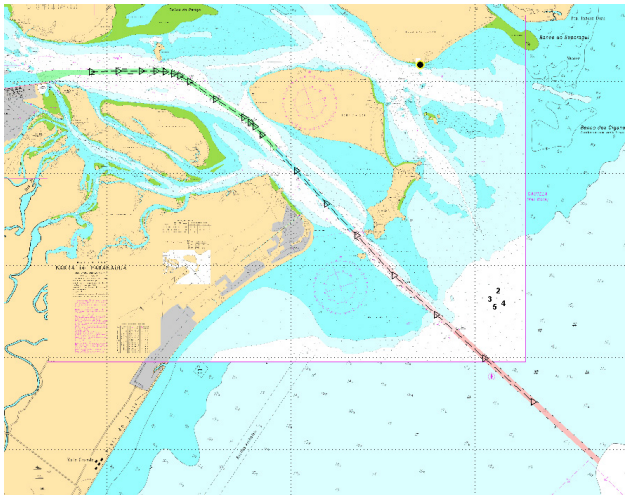

Fig. 6. Access to the Paranaguá port

5. RESULTS

For results validation, we used real test cases, of a ship entering the port of Paranaguá, with the path being given by a sequence of waypoints. The path to be followed can be seen in figure 6. In either side of figure 7, we can see the buoys that indicate the path the ship has to follow, in the real world. The controller was simulated for different environmental conditions, given by table 1.

Table 1. Simulated cases

Case	Sea Current	Wind	Waves
1	Rising tide - 2kn	SE - 21kn	2m - 12.2s - SE
2	Ebbing tide - 2kn	SW - 21kn	2m - 8.8s - S
3	Slack tide - 0kn	NE - 21kn	2m - 9.2s - E
4	Slack tide - 0kn	E - 21kn	2m - 11.1s - ESE
5	Slack tide - 0kn	S - 21kn	2m - 12.3s - SSE

The ship used for these test cases is different than the one used previously. As such, it was necessary to train a new Kalman filter, to obtain new parameters for the dynamic model of the ship. In figure 11, we can see that the path followed by the ship in the 5 cases is almost identical, and strongly coincides with the path defined by the waypoints. In figures 8 to 10, we have the 5 paths overlayed on the nautical chart of the port, considering the size of the ship.

Fig. 7. Example for control situation

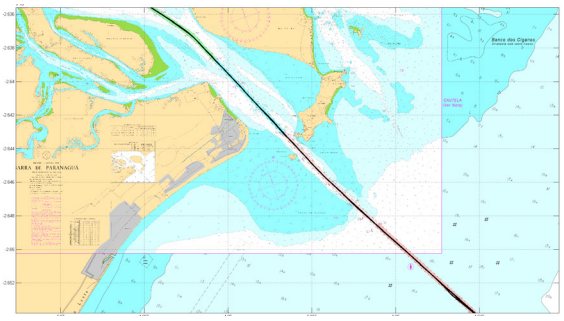

Fig. 8. Initial part of the path

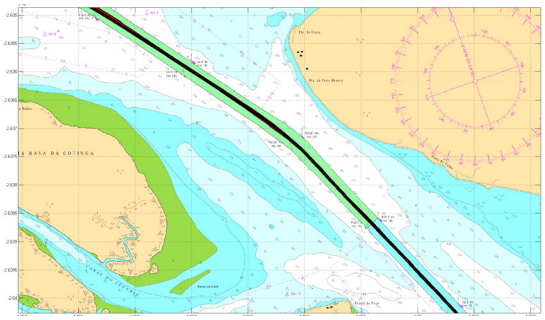

Fig. 9. Middle part of the path

In figure 12, we can see the input given to the simulator by the controller, and the tracking error over the simulation. The large values in tracking error are the points where the LOS system switches the goal, so the system still hasn't had a chance to respond. If the waypoints are too close apart, we can have an abrupt change in desired angle, which leads to the large spikes in the curved part of the path.

The inputs are very oscillatory, and even though they are able to control the system correctly, in some points of the path the input ends up varying, between the maximum and the minimum values for the actuator (without saturation). This could lead to instability, waste of energy and rapid degradation of the actuator. This could be improved by better tuning of the controller parameters, in order to minimize the absolute values of the input and its derivative, so that the controller would get a better result, with a smoother action.

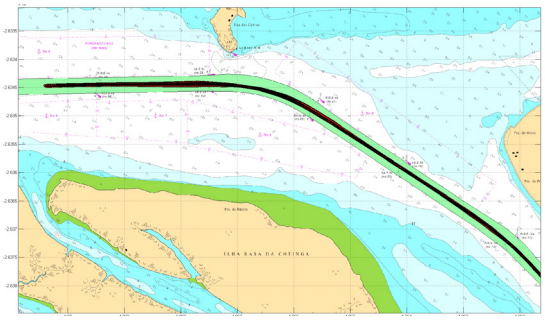

Fig. 10. Final part of the path

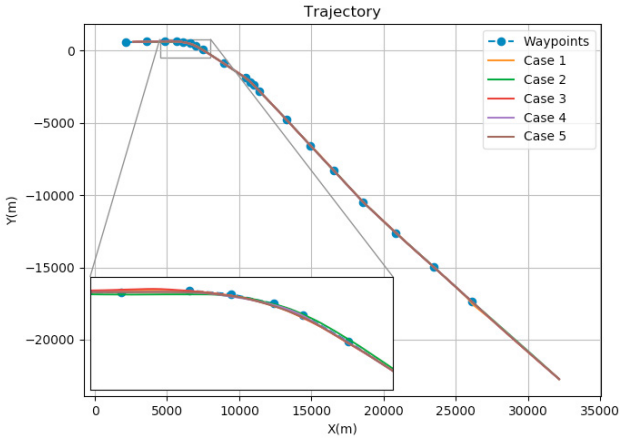

Fig. 11. Paths followed in all test cases

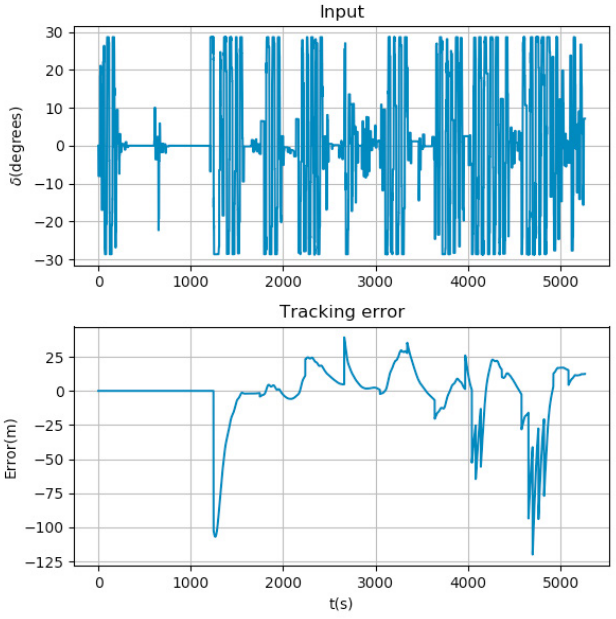

Fig. 12. Input and tracking error for case 1 over simulation

6. DISCUSSION

In this project, we implemented a ship controller using Model Predictive Control. Using LOS, we optimized the desired heading to avoid sudden changes in goal and minimize overshoot when arriving at a waypoint. A Kalman Filter was also implemented, capable of obtaining the necessary parameters for the MPC model. Using the PyDyna simulator, implemented in a Python module, we were able to obtain parameters for the controller and validate our results.

The next step would be to fine tune the parameters of the Kalman filter and the controller, in order to get smoother control inputs and a more stable system identification. Besides that, we could also implement a Collision Avoidance algorithm, capable of generating new waypoints and the returning to the original path, in case of an obstacle on the way.

Generally, the controller was capable of optimizing the inputs of the system, to follow a well defined path, with low error, being able of performing well even in riskier situations when the dynamic of the system changes, such as the entrance of the port, in different environmental conditions. At the end of the project, we had a satisfactory performance, with a system capable of controlling a ship autonomously, in real time.

REFERENCES

Annamalai, A. (2012). A review of model predictive control and closed loop system identification for design of an autopilot for uninhabited surface vehicles. *Technical Report*.

Fossen, T.I. (2011). *Handbook of marine craft hydrodynamics and motion control.pdf*.

Karimi, H.R. and Lu, Y. (2021). Guidance and control methodologies for marine vehicles: A survey. *Control Engineering Practice*, 111(March), 104785. doi:10.1016/j.conengprac.2021.104785. URL `https://doi.org/10.1016/j.conengprac.2021.104785`.

Li, Z., Sun, J., and Oh, S. (2009). Path following for marine surface vessels with rudder and roll constraints: An MPC approach. *Proceedings of the American Control Conference*, 3611–3616. doi:10.1109/ACC.2009.5160302.

Liu, C., Li, C., and Li, W. (2020). Computationally efficient MPC for path following of underactuated marine vessels using projection neural network. *Neural Computing and Applications*, 32(11), 7455–7464. doi:10.1007/s00521-019-04273-y. URL `https://doi.org/10.1007/s00521-019-04273-y`.

Oh, S.R. and Sun, J. (2010). Path following of underactuated marine surface vessels using line-of-sight based model predictive control. *Ocean Engineering*, 37(2-3), 289–295. doi:10.1016/j.oceaneng.2009.10.004. URL `http://dx.doi.org/10.1016/j.oceaneng.2009.10.004`.

Perera, L.P., Oliveira, P., and Soares, G.C. (2015). System Identification of Nonlinear Vessel Steering. *Journal of Offshore Mechanics and Arctic Engineering*, 137(3), 1–7. doi:10.1115/1.4029826.

Siramdasu, Y. and Fahimi, F. (2013). Incorporating input saturation for surface vessel control with experiments. *Control and Intelligent Systems*, 41(1), 49–55. doi:10.2316/Journal.201.2013.1.201-2442.

Skjetne, R., Smogeli, Ø.N., and Fossen, T.I. (2004). A Nonlinear Ship Manoeuvering Model: Identification and adaptive control with experiments for a model ship. *Modeling, Identification and Control*, 25(1), 3–27. doi:10.4173/mic.2004.1.1.

Tannuri, E.A., Rateiro, F., Fucatu, C.H., Ferreira, M.D., Masetti, I.Q., and Nishimoto, K. (2014). Modular Mathematical Model for a Low-Speed Maneuvering Simulator. doi:10.1115/OMAE2014-24414. URL https://doi.org/10.1115/OMAE2014-24414.

Zeng, Z.H., Zou, Z.J., Wang, Z.H., Wang, J.Q., and Others (2019). Path Following of Underactuated Marine Vehicles Based on Model Predictive Control. In *The 29th International Ocean and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Zhang, J., Sun, T., and Liu, Z. (2017). Robust model predictive control for path-following of underactuated surface vessels with roll constraints. *Ocean Engineering*, 143(November 2015), 125–132. doi:10.1016/j.oceaneng.2017.07.057. URL http://dx.doi.org/10.1016/j.oceaneng.2017.07.057.

Zheng, H., Negenborn, R.R., and Lodewijks, G. (2014). Trajectory tracking of autonomous vessels using model predictive control. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 19(3), 8812–8818. doi:10.3182/20140824-6-ZA-1003.00767. URL http://dx.doi.org/10.3182/20140824-6-ZA-1003.00767.