Fakultät für Informatik
Lehrstuhl für Echtzeitsysteme und Robotik

Benchmarking Marine Motion Planning

Benedikt Pfleiderer & Fabian Thomas-Barein

Practical Course *Motion Planning for Autonomous Vehicles* WS 2020/2021

Advisor:	M.Sc. Hanna Krasowski
Supervisor:	Prof. Dr.-Ing. Matthias Althoff
Submission:	14. February 2021

Benchmarking Marine Motion Planning

Benedikt Pfleiderer, Fabian Thomas-Barein
Technische Universität München
Email: benedikt.pfleiderer@tum.de, fabian.thomas-barein@tum.de

***Abstract*—The following paper summarizes the development and features of CommonOcean. CommonOcean is a benchmarking tool for marine motion planning developed in Python. It has been designed as a subpackage of CommonRoad, which is a collection of benchmarks for motion planning on roads. In addition, CommonRoad includes several tools for motion planning such as a Drivability Checker or a set of search algorithms. With CommonOcean, we now want to transfer benchmarking scenarios and related tools to the maritime sector. The presented project features reading, writing and plotting of CommonOcean scenarios. In addition, a search-based motion planner which can solve planning problems using motion primitives was developed. The motion primitives are created with the help of a point mass planner. The result is a functioning benchmarking tool that meets all requirements and still has extension possibilities, such as the reading of nautical charts and the plotting of AIS data. Due to a structure similar to its parent project, the benchmarking tool is easy to understand with prior experience in working with CommonRoad.**

Fig. 1. CommonOcean logo

I. Introduction

With the increasing globalization the demand for the transportation of goods increases as well. This leads to a rise in the density of maritime traffic since 90% of all goods on the world market are transported via cargo vessels [2]. With that amount of traffic it is not surprising, that accidents occur frequently [1] and with the large dimensions of today's cargo vessel, the stakes for disasters are high. It is not uncommon that propulsion fuel or even crude oil leaks out of damaged ships. Resulting natural disasters are usually far worse than the actual shipping accident.

With 75% up to 96% of accidents being caused by human error [3], automation has a large potential to drop the number of accidents significantly. Especially in view of the fact that the development of autonomous cars is already so far advanced, there is still a lot of potential and need to catch up in the maritime sector. To drive the development forward, a benchmarking tool is needed to efficiently evaluate automation algorithms for vessels with real data.

Additionally it is mandatory for all commercial ships to send out information via the Autonomous Identification System (AIS), which can be received by other vessels in the area. This leads to a high data density of vessel meta data, which can be perfectly used for creating benchmarks.

II. Implementation

In the following section we discuss the implementation of CommonOcean. Starting with a discussion about certain modelling decisions, an overview of the class structure follows. Then we continue with a summary of which packages originally developed for CommonRoad [6] had to be modified to make them work.

A. Modelling Decisions

We decided to model CommonOcean as a subpackage of CommonRoad. Thus, keeping the structure as modular as possible at the same time. Simultaneously, however, we had to formalize the structure of a maritime scenario into the same structure as a CommonRoad scenario. On the one hand, a lot of existing structure can be kept, but some parts have to be adapted. Besides the fact that dynamic objects had to be extended by a third dimension, since we now also have to consider collisions with the seabed, we had to break up the structure of fixed lanelets on roads and find an abstraction for waters.

In Figure 2, a typical fairway through a strait is depicted. This is the standard representation of a sea chart: The yellow and grey parts are landmasses, connected by a bridge. The dark blue areas are areas of shallow water up to a depth of five meters, the lighter blue is between five and ten meters and the white area, visible in the lower right corner, stands for every depth deeper than ten meters.

The fairway between the landmasses is on both sides confined by buoys, leading the traffic through the shallow parts and under the bridge.

We modeled the fairway as the equivalent of a lanelet in CommonRoad, with the open water on either side being the successor and predecessor. This is emphasized in Figure 2 by the red lines, enclosing the fairway. In the open water we

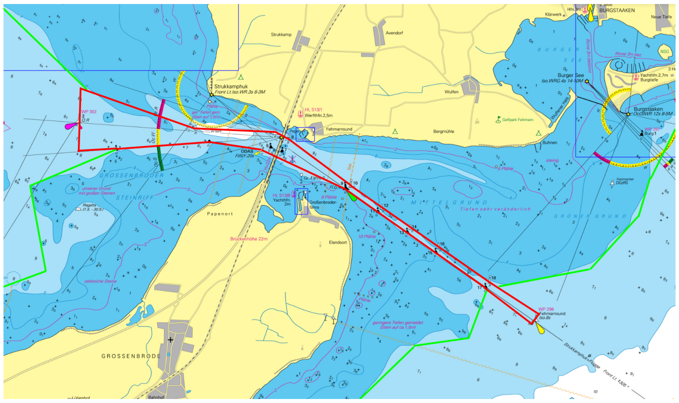

Fig. 2. Nautical charts with waters network

had to make sure, that we exclude the shallow parts, where most vessels are not able to sail. This is done by introducing drafts as a new type of static obstacle. In the Figure 2, this is exemplary shown by the green lines around the depth marker of five meters.

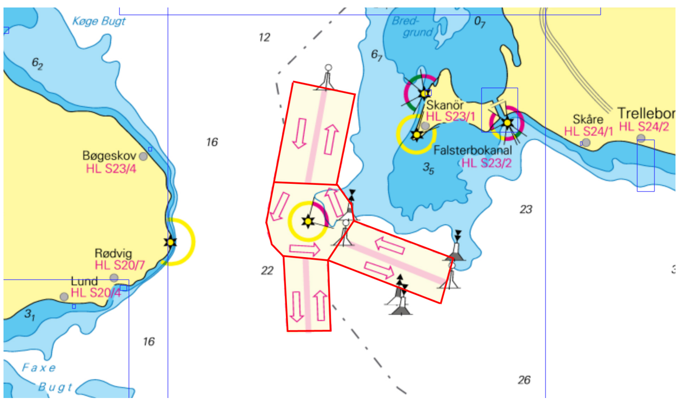

Fig. 3. Nautical charts with a traffic separation zone

Another structured traffic confinement ships encounter on the water is the traffic separation zone, which has stricter rules than the regular fairway. As shown by the arrows, seen in Figure 3, vessels are only allowed to travel in one direction on either side of the fairways, leading towards the roundabout in the middle. Further more it is not allowed to cross a traffic separation zone except perpendicular and on the shortest path.

Due to the rules of a regular fairway, which state that you have to keep on the right side, we decided to model the traffic separation zone in this example as four separate waters.We saw no advantage in increasing the complexity, since we would obey these rules nonetheless.

B. Classes and Objects

In the following description of the structure, we will draw a parallel to the structure of CommonRoad, so that the structure of our project is better understandable. A simplified basic structure of the project can also be seen in Figure 5 as UML diagram.

The benchmark object is always a Scenario object, as in CommonRoad. The main properties are a network of waters, lists with its static and dynamic obstacles and a list of drafts. The WatersNetwork object function analogue to LaneletNetwork and describes a network of waters, which are adjacent to each other. In the section on modeling decisions II-A we have already introduced how individual waters are abstracted from nautical charts. A single waters has left and right vertices, a type, metadata such an unique id and stores especially Traffic signs, which are located on a particular waters.

Static and dynamic obstacles are very similar to CommonRoad. Its biggest change is the addition of a third dimension in size to prevent collisions with the bottom of the sea.

Drafts or shallows are areas on the map that are not very deep and therefore should not be crossed, especially by large vessels, to prevent the boat from grounding. Real depth values are not yet taken from maps, but draft depths are set to static 10 meters, and are not traversed by the motion planner with any vessel. Draft objects do not have an analog object in CommonRoad, but are not very complex either. They are described by a Shape object and an associated depth marker.

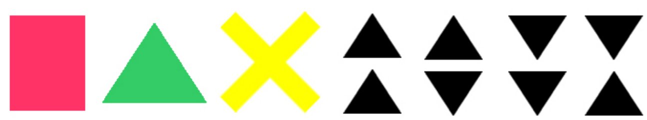

Fig. 4. Overview of supported traffic signs

Traffic signs are again very analog to the well-known CommonRoad traffic signs. We changed types, kinds and plotting images, so that Maritime traffic signs can be plotted correctly. Currently supported and included in the project are lateral marks as green and red buoys and cardinal marks, which are used in maritime pilotage to indicate the position of a hazard and the direction of safe water. In addition, we also support the plotting of a special mark. Figure 4 shows a summarized overview of all included traffic signs.

C. Drivability Checker

The CommonRoad Drivability Checker toolbox unifies the validation of collision avoidance, kinematic feasibility, and road-compliance to ensure the drivability of planned motions for autonomous vehicles in order to simplify the development and validation of motion planning algorithms.[5] It is compatible with the CommonRoad benchmark suite, which additionally facilitates and drastically reduces the effort of the development of motion planning algorithms.

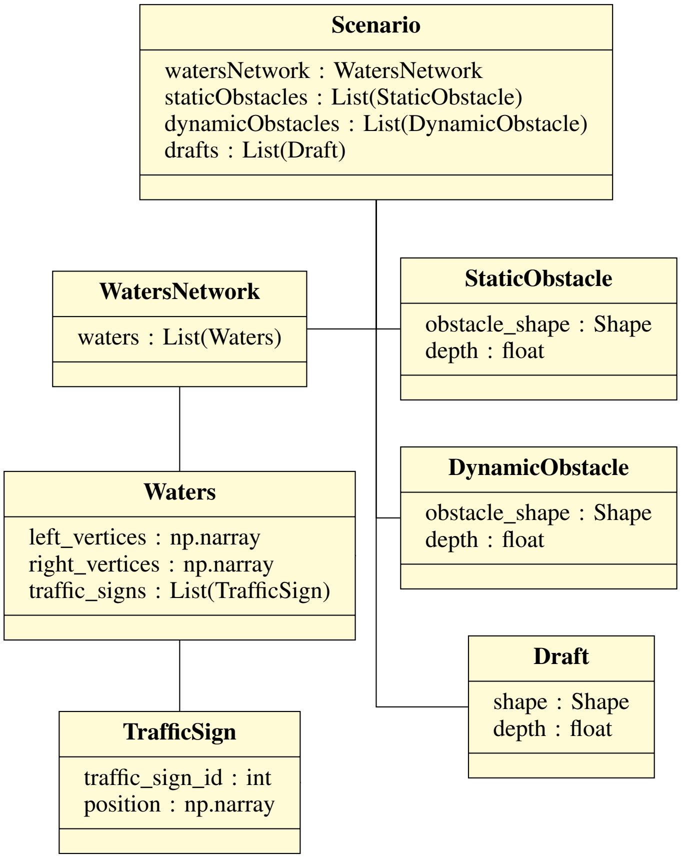

Fig. 5. UML Diagram of CommonOcean Scenario

Due to the similar structure of CommonOcean to CommonRoad, the functions from the Drivability Checker could be easily utilized. We included our remodeled static and dynamic objects as well as drafts so that collision objects can be generated for them as well. The collision objects are needed so that the search algorithm can later solve a planning problem, without colliding with objects.

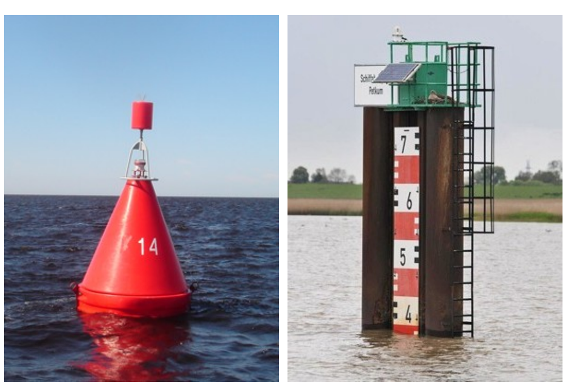

Fig. 6. Different types of traffic signs

Particularly noteworthy is the handling of traffic signs. Each of the traffic signs gets a matching static obstacle automatically generated based on the type of the traffic sign. We primarily distinguish between floating buoys and static poles as seen in Figure 6. Static poles are, as the name suggests, static and only need a collision object that is approximately the size of the pole. Buoys, on the other hand, are attached to the bottom of the sea and can swing, so they need a collision object larger than their spatial extension in order to prevent collision. The collision objects for the traffic signs are modelled by circles. The size of the collision circles is currently implemented statically, but could be adjusted dynamically in a later version, based on water depth and stream. Figure 7 shows an overview of all collision objects. Marked in red are static obstacles in different sizes. Dark blue are dynamic obstacles and light blue is the collision box of a draft in this scenario.

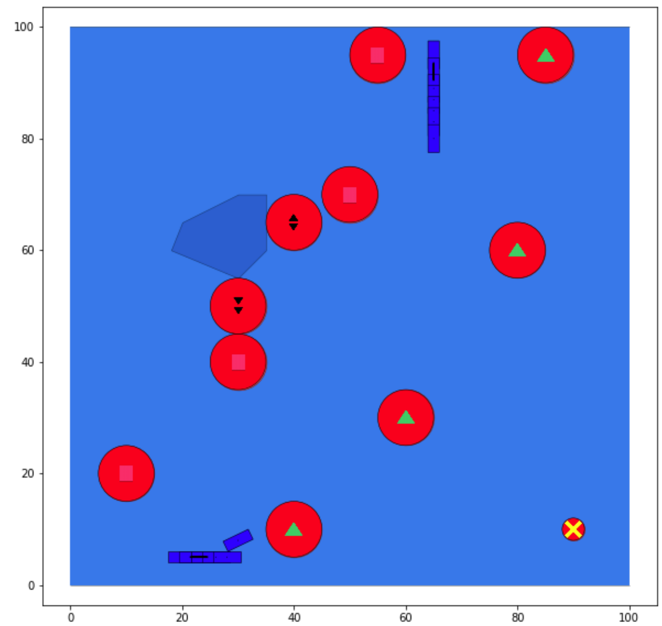

Fig. 7. Created collision objects

D. Solving Planning Problems

In the next and last step we want to solve plannings problems in our scenario. For this we will first introduce the needed motion primitives and then about the actual search algorithms.

1) Motion Primitives: Motion primitives in CommonRoad Search, as well as in our CommonOcean Search, describe trajectories that are to be evaluated by the search algorithm to generate a path from the initial state to the goal state and are also technically and physically reasonable and executable for the ego vessel. Once all the desired primitives have been generated, several of them are combined into a Maneuver Automaton for the search algorithm. The Maneuver Automaton is a class, witch is able to hold and manipulate a set of motion primitives for the motion planning by a search algorithm. A motion primitive is very similar to a single trajectory that

describes a path. There are also developments of Decision Support Systems (DSS), which can give maneuver suggestions to the captain of a ship, based on a large database with sets of trajectories.[4]
CommonRoad provides a lot prefabricated motion primitives and a generator, but all this is designed for the physical and technical properties of today's cars and is therefore not suitable for our project, in which we want to maneuver inert vessels. In order to better model the behavior of ships, we used a 2D point mass planner to generate motion primitives.
This planner takes into account the dimensions of a vessel, since the maneuverability diminishes with increasing size and weight. To realistically model the behaviour of large vessels, also the current speed plays an important role. With lower speeds, ships have difficulties to reliably maneuver in confined spaces but are able to perform relatively sharp turns. But with higher speeds, the rate, with which the vessel is able to turn, decreases again.
All these characteristics have to be considered in order to make a feasible prediction, how the ego vessel will be able to maneuver.

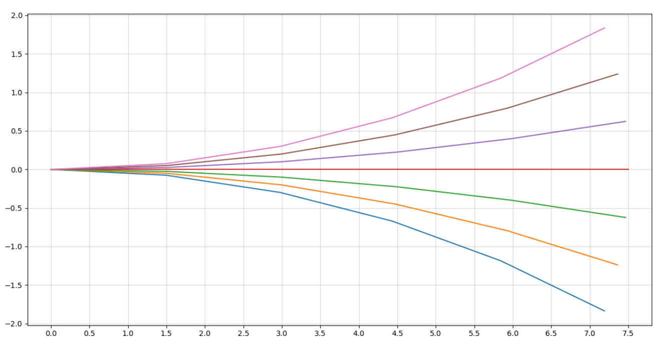

Fig. 8. Motion primitives, generated by the point mass planner with the vessel size of 7.7m length and 2.5m width at a static speed of 15m/s

The Figure 8 depicts a set of motion primitives, generated with the 2D point mass planner. The parameters are set for a very small vessel, since we used it for a small exemplary scenario. This planner returns a set of trajectories, the ego vessel will be able to follow and these are passed to the previously mentioned motion primitive generator from CommonRoad to merge the individual trajectories into a set of motion primitives with the format that CommonOcean Search requires.

2) CommonOcean Search: In order to actually solve a planning problem, we adapted CommonRoad Search for our use to CommonOcean Search. CommonOcean Search takes the provided motion primitives and appends any of the trajectories on the last explored path. It thereby checks for every step, if the path collides with a collision object or the waters boundaries. If so the exploration of the current path is stopped.

CommonOcean Search supports the same search algorithms as CommonRoad Search but only the informed search algorithms provide enough performance for our use cases. This is due to the vast non occupied area that is the open water. In the CommonRoad environment, the search algorithm is mostly restricted to one lanelet. But as we discussed in the modelling decisions, we diverted from this approach and any uninformed search algorithm generates so many possible paths, that the resulting performance is not reasonable.

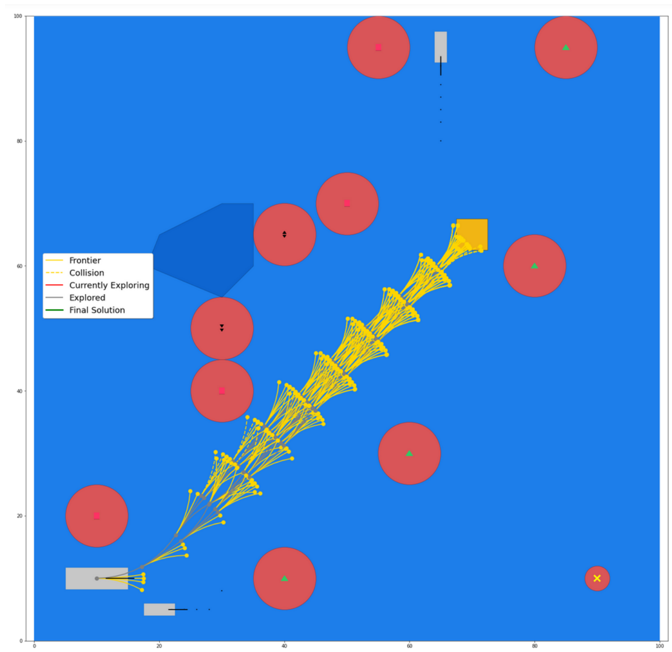

Fig. 9. A* search algorithm with a simpler planning problem

Another problem for a good performance of the search are not well selected motion primitives. We found that too little distance between the end nodes of the trajectories pose a challenge to the search algorithm, due to the similar positions and therefore very similar distance to the goal state. This prevents the heuristic function from the informed search algorithm to make sensible predictions on the distance of the currently explored path. This leads to a lot of unnecessary redundancy. The implemented solution for this problem is using motion primitives with a large enough distance between the end nodes of the motion primitives. This way the search algorithm recognizes faster, which path is reasonable to explore further. However this reduces the possible positions which can be occupied by the ego vessel, but that is acceptable for striking a balance between the higher precision from the increased number of positions as well as the much better performance. This can be seen in the Figure 9 where a simple planning problem is solved by the A* search algorithm. The scenario is the same, we introduced for the CommonRoad Drivability Checker except for the planning problem. The planning problem is very simple with a direct route from the initial state to the goal state and hardly any collisions. But the lack of collisions also means that the search algorithm cannot discard of any paths and due to the little difference in the distance of the paths to the goal state, every path has to be pursued.

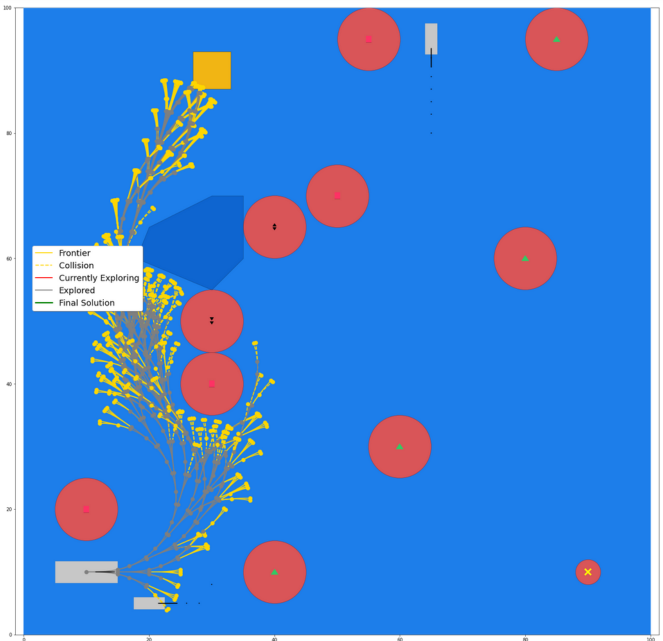

Fig. 10. A* search algorithm with a more difficult planning problem

In the Figure 10 another planning problem with the result of an A* search algorithm is depicted. The initial state is in the bottom left corner and the goal state is the yellow square in the upper left corner. The search algorithm then finds the shortest path from the initial state to the goal state without any collisions. In this example the goal state is located behind collision objects. The collision objects reduce the possible paths from the initial state to the goal state and thereby increases the performance.

III. Results

In the following we summarize our results from the development of CommonOcean, which is a benchmarking tool for motion planning on waters.

We have created CommonOcean, as a subpackage for CommonRoad to stay as modular as possible for further projects. On one hand, CommonOcean can be used as a benchmark tool to read, write and plot scenarios and planning problems. In addition, it can solve planning problems and find solutions through an adapted version of CommonRoad Search, called CommonOcean Search. Lastly, for other students to easily continue on this project, we made an effort to have a good code structure and documentation.

There are still some points left, which can be well taken up again for future work on this topic as reading and plotting of real maps. OpenSeaMaps, which is a subproject of OpenStreetMap, provides very good map data, which can be used to generate realistic maps for motion planning. In Figure 11 you can see a section of the harbour Burgtiefe on the German island of Fehmarn. For CommonRoad a python tool for reading OpenStreetMap data already exists. This map tool could be easily adapted to fit the CommonOcean scheme. Furthermore, the project could be extended by adding a converter from AIS to CommonOcean, taking advantage of the information provided. AIS features information such as speed and heading and is mandatory for most large and commercial ships, which is why there is a good data density. Further, using AIS data and sea maps, an automatic benchmark generator could be implemented. Finally, the performance of CommonOcean could be improved as well. CommonRoad has very strict boundaries with its fixed lanelets, which help a lot to keep the computation time low when solving a planning problem. In a CommonOcean scenario with a lot of open sea, however, we sometimes come up against computational capacities, if we do not use an informed search with suitable heuristics. This could be counteracted by limiting the path search to the current fairway only, using the red and green buoys as boundaries.

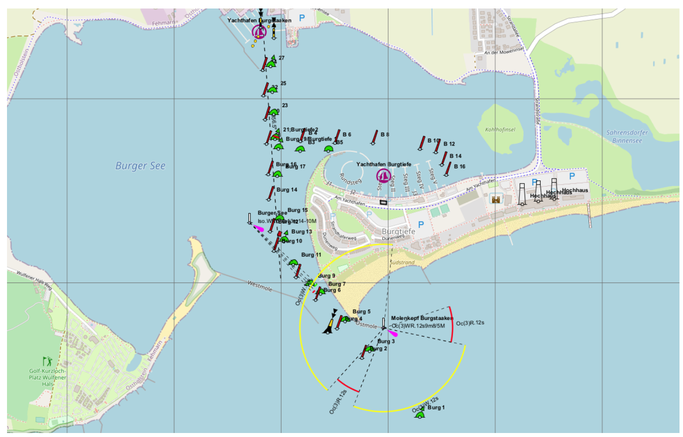

Fig. 11. OpenSeaMap excerpt of Burgtiefe harbor

IV. Conclusion

In summary, maritime applications offer a great deal of potential for automating maneuvers and entire routes. Especially the availability and reliability of the AIS system is a good data basis for autonomous maneuvers, which is far ahead of car traffic. It is even more surprising that so many accidents in the oceans of this world are still caused by human error. It would be more than reasonable to invest money and time in the development of new and safe systems in a form of transport that is so cheap on long distances for the transport of goods. With CommonOcean we have the foundation for a benchmarking tool, so that research on such innovative routing planners can be done easily with real data are comparable. CommonOcean is very similar to the parent project CommonRoad, which has been used for benchmarking on the road for years. The similar structure offers many advantages in terms of usability and thus many parts of CommonRoad could be reused.
CommonOcean still has a lot of potential to be extended

with further functions and the development has also shown which parts of the code structure are used by both projects. Therefore, one idea is to create a common framework from which both application frameworks can be inherited and even further applications like aerial motion planning can be easily developed.

REFERENCES

[1] Guo, Siyu, et al. "An autonomous path planning model for unmanned ships based on deep reinforcement learning." Sensors 20.2 (2020): 426.

[2] Tu, Enmei, et al. "Exploiting AIS data for intelligent maritime navigation: A comprehensive survey from data to methodology." IEEE Transactions on Intelligent Transportation Systems 19.5 (2017): 1559-1582.

[3] A. Harati-Mokhtari, A. Wall, P. Brooks, and J. Wang, "Automatic identification system (AIS): Data reliability and human error implications," J. Navigat., vol. 60, no. 3, pp. 373-389, 2007.

[4] Lazarowska, Agnieszka. "A trajectory base method for ship's safe path planning." Procedia computer science 96 (2016): 1022-1031.

[5] Pek, Christian, et al. "CommonRoad Drivability Checker: Simplifying the Development and Validation of Motion Planning Algorithms." 2020 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2020.

[6] Althoff, Matthias, Markus Koschi, and Stefanie Manzinger. "CommonRoad: Composable benchmarks for motion planning on roads." 2017 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2017.