System Architecture and Implementation of Global Vessel Path Planning Based on ROS2 Framework

Enio Krizman[1], Nadir Kapetanović[2], Đula Nađ[2]

***Abstract*— This paper presents a system architecture for global path planning developed within the ROS2 framework for Uncrewed Surface Vehicles (USVs). The system is structured into three key stages: generating a cost map through geographic feature extraction, managing start and goal coordinates, and executing path planning using the D* Lite algorithm with interpolation techniques. The cost map, based on OpenStreetMap data, provides a detailed representation of the operational environment, while RViz2 is used for interactive setting and visualization of start and goal coordinates. To address the sharp path transitions produced by the D* Lite algorithm, this paper introduces a path interpolation model that consists of coordinate resampling and curve fitting techniques. The hypothesis is that, with correctly configured cost settings and interpolation parameters, these methods can support real-world vessel path planning. Simulation tests on navigational maps validate this approach, demonstrating its suitability for further development and integration into USV system architectures.**

***Index Terms*- Global path planning, ROS2 system architecture, cost map, D* Lite, interpolation**

I. INTRODUCTION

In recent years, the maritime industry has increasingly focused on developing Uncrewed Surface Vehicles (USVs) with varying levels of autonomy for commercial, scientific, and military applications. USVs can enhance situational awareness by processing sensor data and utilizing advancements in Guidance, Navigation, and Control (GNC) systems, making them an interesting subject for exploring their capabilities in various applications.

Traditional crewed vessels have used semi-autonomous systems for management, forecasting, and planning. However, these systems were not designed for uncrewed vessels, which require rapid data processing and a modular communication architecture that supports various hardware and software components. Therefore, their implementation needs to be reevaluated. A key example is vessel path planning systems, many of which have commercial versions. The issue with these commercial systems is that they are tied to the manufacturer's device, limiting access to and control over data processing and communication tools. Additionally, upgrading planning algorithms to accommodate different optimization parameters based on mission objectives is nearly impossible. Due to these limitations, a vessel path planning architecture based on the ROS2 framework [1] is being considered for its flexibility in upgrading and adapting to various requirements. It is also gaining broader acceptance in autonomous system solutions, making it a favorable choice for future USV systems.

The ROS2 framework includes the ROS2 Navigation Stack (NAV2), which is used for controlling and navigating mobile robots in complex environments [2]. However, the current version of the NAV2 system does not support route planning for vessels. While the existing system could be adapted for this purpose, it was decided to develop a new system from the ground up, incorporating certain concepts from existing solutions. Several scientific papers address geographic feature extraction [3], the creation of cost maps based on different parameters [4], and the implementation of algorithms for vessel path planning [5]. However, to the best of the author's knowledge, no known work represents a complete ROS2 system architecture for vessel path planning. The NAV2 work [2] utilizes some similar concepts, but its focus is not just on path planning implementation. It addresses a more general navigation architecture that includes vehicle control and local path planning in dynamic environments.

Therefore, a software architecture for a global vessel path planning system is proposed, built on a modular communication topology within the ROS2 framework. Vessel path planning is seen as part of a broader system architecture with multiple components, although it can also operate as a standalone solution. A cost map generation process is introduced, based on feature extraction from OpenStreetMap (OSM) data, alongside a path planning concept utilizing D* Lite interpolation with coordinate resampling and curve fitting techniques that can be applied for different vessel types. The system components are designed to function as independent modules, allowing them to be further developed or replaced with more advanced versions without affecting the overall architecture's functionality.

The paper is organized as follows: In Section II, the architecture of the global path planning system is described, with a focus on key stages, distinct software components, communication technology, and coordinate system conversion rules. Section III explains the creation of cost maps. Section IV covers data visualization and the setting of start and goal coordinates. The adaptation of the D* Lite algorithm and the interpolation techniques are elaborated in Section V. Finally, the results tested on navigational area maps are presented in Section VI.

*This research was funded by the Croatian National Recovery and Resilience funded project Smart Blue Tourism, G.A. No. NPOO.C1.6.R1-I2.01-V3.0007.

[1]Department of Marine Technology (IMT), Norwegian University of Science and Technology (NTNU), Jonsvannsveien 82, 7050 Trondheim, Norway enio.krizman@ntnu.no

[2]Laboratory for Underwater Systems and Technologies (LABUST), University of Zagreb Faculty of Electrical Engineering and Computing (FER), Unska 3, 10000 Zagreb, Croatia nadir.kapetanovic@fer.hr, dula.nad@fer.hr

II. ROS2-BASED SYSTEM ARCHITECTURE

A. System Architecture

Developing a functional software system architecture for global vessel path planning is a complex task, involving several logical stages, each presented with distinct software components known as packages within the ROS2 framework. The system architecture is divided into three critical stages, as illustrated by the block diagram in Fig. 1.

The first stage involves creating a cost map based on geographic feature extraction from OpenStreetMap data [6]. This is handled by the Map-Making package, which integrates the geographic maps into the ROS2 framework using a ROS2 publisher.

The second stage includes setting the start and goal coordinates using RViz2 visualization tools or geographical data, managed by the Start-Goal Management package. This package not only handles the setup and visualization of the coordinates but also publishes this information within the ROS2 framework using the action client-server communication mechanism [7]. Additionally, it manages data conversion for visualization purposes. The focus in the following sections will be on describing the functionality of the Map-Making and Path Planning packages due to the concepts they introduce, while the Start-Goal Management package will be briefly covered.

The final stage focuses on path planning, where the Path-Planning package adapts the D* Lite algorithm and applies interpolation techniques to generate a smooth and feasible path. This package facilitates communication between the client and server, exchanging waypoint information and performing the necessary path adjustments.

It is important to note that each part of the path planning architecture has its own unique approach and can be further developed as an independent module. Additionally, this represents only one possible solution for each stage of the process and can be upgraded or replaced in the future without interfering with the functionality of other system components. This flexibility is made possible by utilizing the coordinate systems conversion described in II-B and the communication topology described in Section II-C.

B. Coordinate Systems

Coordinate systems are essential in mathematics, physics, geography, and other scientific disciplines, as they allow for the precise definition of point positions in space. To fully grasp system architecture concepts, it is important to first explain the terms global and local coordinate systems, as well as the conversion process between them.

The Global Coordinate System is defined as a geographic coordinate system used to represent Earth's surface features. This paper adopts a global, earth-fixed geographical coordinate system based on the Pseudo-Mercator projection (EPSG:3857) [8], which is widely employed in digital mapping and geographic information systems (GIS) due to its effectiveness in displaying geographic data on flat maps.

The Local Coordinate System is derived from the global system by converting geographic coordinates into a metric scale, starting with the extraction of a portion of the global system for localized data processing. Depending on the specific task, data processing may involve operations such as image feature analysis for coastline detection, coordinate alignment into a merged map, the creation of a cost map for path planning, or adapting geographic coordinates for use in visualization tools.

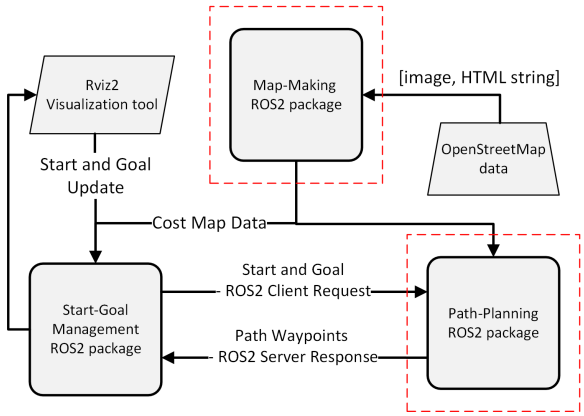


Fig. 1: Block diagram of the ROS2 system architecture. The dotted red lines highlight the main focus of this paper.

The extraction of a portion of the global coordinate system defines the geographic area by identifying four boundary coordinates, representing the minimum and maximum values for data processing. The conversion process transforms geographic coordinates from degrees to radians and applies the Haversine formula to compute the great-circle distance in meters [9]. Geographic coordinates are converted into a local system using a step-size sampling parameter, aligning and rounding values. Any data loss due to coordinate sampling is mitigated through curve fitting, which helps to minimize information loss for path planning purposes.

C. Communication Topology

The system's communication topology is based on ROS2 communication principles, where different packages exchange information using geographic coordinates from the global coordinate system. In simple terms, each ROS2 package receives geographic coordinate information from another package. This data is converted from the global to the local coordinate system, processed according to the package's specific function, and then converted back to the global coordinate system.

Information is exchanged between ROS2 nodes within packages. The publisher-subscriber communication mechanism ensures message exchange, which is used to publish information about the cost map, visualization data, and start-goal coordinate updates. Additionally, the server-client mechanism employs actions to manage the vessel's path planning process. The start and goal coordinates are submitted in the request field, while raw and interpolated path information is provided in the feedback field. Communication data is illustrated in Fig. 1, with arrows indicating the flow between blocks.

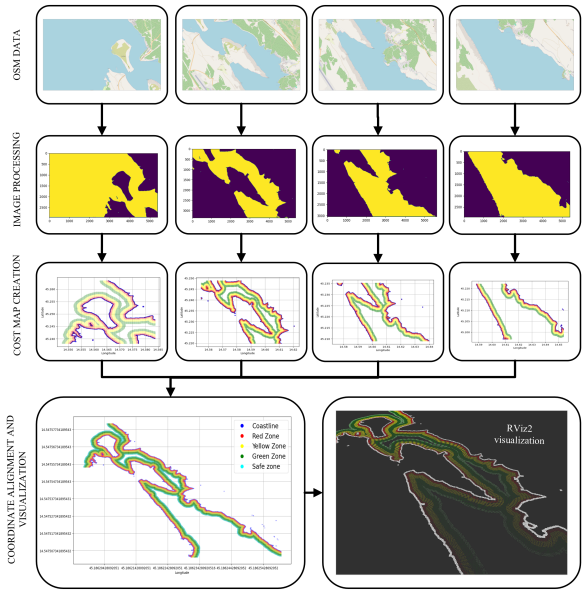


Fig. 2: Block diagram illustrating the cost map creation process. The first stage involves fetching data from OSM. In the second stage, the coastline is extracted using image processing methods. The third stage constructs a cost map with distance-based red, yellow, green, and safe zones. In the final stage, the cost maps are merged by aligning the coordinates, and the result is visualized using RViz2.

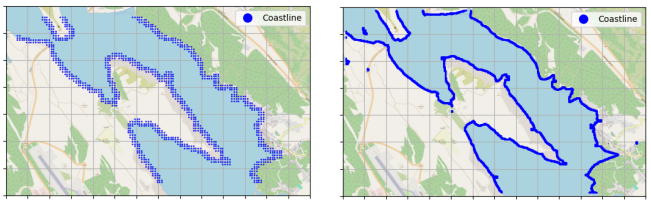
III. COST MAP CONSTRUCTION

The initial step in developing a global vessel path planning system is the creation of a cost map based on geographical feature extraction. To accomplish this, the authors developed the Map-Making ROS2 package, which processes data from OpenStreetMap [6] due to the unavailability of digital nautical maps. This process is illustrated in Fig. 2. The package generates a map containing coastal coordinates and distance zones, which are used as a cost map in the path planning process with adjustable cost values. However, additional features, such as seabed morphology and danger zones, will be integrated in future phases of the project.

A. Downloading and Processing Data

The OSM website offers various zoom levels, with the 300-meter elevation providing a balance between detail and manageability for data processing. The data is downloaded by navigating to OpenStreetMap, selecting the desired area, and obtaining the PNG map image and HTML snippet which is labeled as 'osm data' in Fig. 2.

The HTML snippet contains the boundary coordinates that define the geographic area's edges. These coordinates are used to create a local coordinate system. The scaling principle of one pixel per square meter for the fetched image ensures accurate alignment between pixel positions and real-world geographic coordinates, facilitating precise feature extraction.



(a) Coastline coordinates sampled with a larger step.

(b) Coastline coordinates sampled with a smaller step.

Fig. 3: Comparison of coastline sampling at different grid sizes.

B. Coastal Coordinate Extraction

Coastal coordinates and distance zones are identified by converting the scaled image from RGB to HSV color space for better color distinction, as labeled 'image processing' in Fig. 2. The sea is detected using an HSV range of [80, 60, 60] to [120, 255, 255], generating a binary mask where sea pixels are white (255) and all others are black (0). To remove topological names over the sea, binary closing is applied, ensuring that the closing matrix is small enough to preserve small geographic features such as islands.

Coastal coordinates are identified by detecting transitions from 0 to 255 values in the binary mask, which marks the coastline. These coordinates are sampled based on the step-size parameter grid_size, where only coordinates in the local coordinate system that are multiples of grid_size are examined to manage computational complexity. Selecting an appropriate parameter is crucial, as larger values can result in loss of detail (Fig. 3a). To balance computational complexity and data accuracy, a grid_size of 10 meters was chosen, offering a compromise between efficiency and the preservation of detail (Fig. 3b). Furthermore, determining coordinates from processed images can lead to errors, such as edge pixels being misidentified as coastline. To fix this, coordinates within 50 pixels of the edges are discarded, and bridges are corrected by adjusting their color to match the sea in image editing software. These methods ensure accurate coastline representation.

C. Creation of a Merged Cost Map

The final step in creating a cost map involves defining distance zones and aligning coordinates from image processing to form a cohesive map of geographic features. Croatian maritime regulations [10] provide speed rules based on distance zones from the coast and vessel types, which are used as guidelines in the cost map creation. Each zone is assigned a cost value, adjustable for different path planning optimizations.

The red zone extends up to 50 meters from the coast, where most obstacles and shallow waters are located. The yellow zone spans 50 to 150 meters and presents moderate safety concerns. The green zone is established between 150 and 300 meters, in accordance with Article 48, which limits vessel speed in areas less than 300 meters. Lastly, the safe zone spans 300 to 350 meters and is introduced for path

planning optimization, though it is not governed by any specific regulations. These zones are determined based on the distances from the coastal coordinates, and this process is labeled as 'cost map creation' in Fig. 2. For the path planning process, the cost map values are assigned as follows: infinity for the coastline, 1 for open sea, and adjustable values for red, yellow, green, and safe zones. The cost of the red zone is the highest, decreasing with each zone in the outward direction. This is done to force the path planning algorithm to favor not only the shortest path but more importantly paths that move away from the coast to zones that allow higher surge speed and safer navigation with fewer expected obstacles. Cost assignment can be changed depending on specific characteristics of the vessel and coastline, and (inter)national maritime regulations.

One challenge with OSM is that the zoom level of 300, which provides detailed images, does not cover large enough areas typically required for vessel path planning. Consequently, images from nearby areas must be extracted and merged into a unified representation, as shown in the final part of Fig. 2. Coordinate alignment is crucial in this process, as the extracted coastline coordinates and defined zones from different images depend on boundary coordinates and sampling parameters, which can lead to misalignment in the global representation of larger areas. To address this, a local coordinate system is established by merging coordinate lists, extracting global min/max values, and aligning the coordinates to multiples of grid_size. The final step consists of converting the coordinates back to the global system and publishing the map, which is used as the cost map in the path planning process and for data visualization in RViz2.

IV. START-GOAL MANAGEMENT AND VISUALIZATION

The Start-Goal Management ROS2 package plays a key role in vessel path planning by managing the setting and visualization of start and goal coordinates, geographic maps, and calculated paths. Users can interactively set the start and goal coordinates within a 3D map in RViz2 or manually input them through terminal commands.

Once the coordinates are defined, the system uses an action client-server communication, where the client sends the coordinates to the server, which calculates both interpolated and D* Lite paths. RViz2 is used to visualize the calculated paths offering a detailed view of the vessel's route. An example of this 3D visualization is shown in Fig. 4.

V. PATH PLANNING USING D* LITE AND INETRPOLATION TECHNIQUES

Global path planning, performed prior to a mission using the Path-Planning ROS2 package, determines the vessel's route based on a cost map, helping the vessel navigate safely and avoid obstacles. D* Lite calculates the path by minimizing traversal costs, and interpolation techniques smooth the route, ensuring it is physically feasible for the mission.

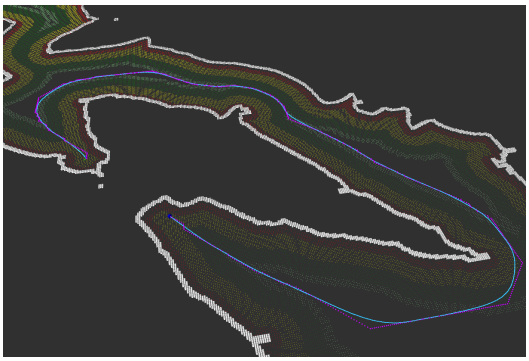


Fig. 4: 3D visualization in the RViz2 tool. The coastline is represented in white, with zones shown in red, yellow, green, and light green (safe zone). The raw path generated by D* Lite is displayed as magenta curves, and the interpolated final path is shown in cyan.

A. D* Lite Algorithm Adaptation

The D* Lite algorithm used in this paper is adapted from the PythonRobotics repository [11], based on the work of Sven Koenig and Maxim Likhachev (2002) [12], with modifications for handling cost maps. Testing various path planning algorithms revealed D* Lite to be the most efficient due to its shorter runtime and ability to function in dynamic environments, making it suitable for future project stages. The process begins with the Map-Making package publishing geographic data that forms a local coordinate system with a parametric cost map. The action client then processes the start and goal coordinates and runs the D* Lite algorithm.

The adaptation of the D* Lite algorithm involves defining the cost based on the zone costs in the cost map and modifying the algorithm's steps for searching the next nodes. The focus was not solely on optimizing D* Lite's performance but rather on integrating it into the overall system architecture. Consequently, the adaptation was limited to working with the cost map without incorporating the vessel's dynamic model. This results in a path that includes sharp turns (Fig. 5.). To ensure the path is feasible, a path interpolation system was developed, as described in the next subsection.

B. Path Interpolation

The D* Lite algorithm from the previous step often generates infeasible vessel paths. Therefore, interpolation techniques, including coordinate resampling and curve fitting, are introduced to avoid sudden turns and ensure a smoother path. The path interpolation process is illustrated in Fig. 5.

The coordinate resampling process uses trigonometric functions, such as Euclidean distance and the cosine rule, to adjust the path's curvature by calculating distances and angles between coordinates. Identifying sharp directional changes is crucial, as vessels have a maximum turning rate and cannot change course quickly. The downsampling rate parameter determines which coordinates from the D* Lite path, along with sharp turn coordinates, are included in the resampling process. For example, with a downsampling rate

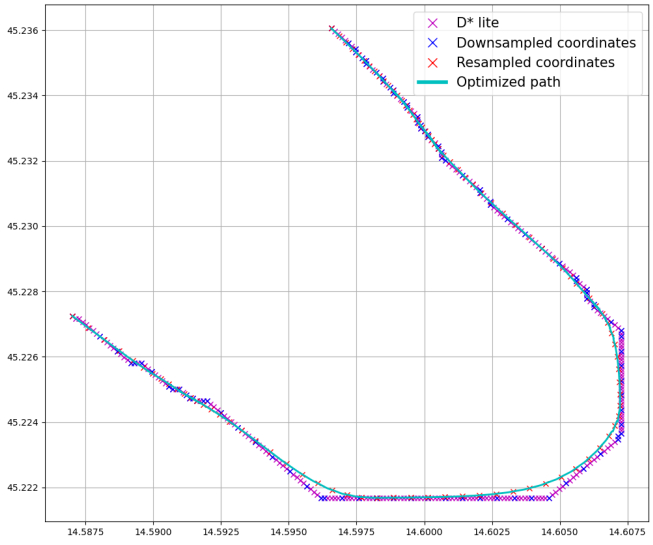


Fig. 5: Result of the path interpolation process.

of five, every fifth coordinate from the D* Lite path is selected. Sharp turn coordinates and downsampled coordinates are used for resampling, as indicated by blue x symbols in Fig. 5. During each iteration, the algorithm checks for angles smaller than 150 degrees formed by three consecutive coordinates. If such an angle is detected, the middle coordinate is resampled to a position between the first and third coordinates to reduce curvature. This process continues until all angles exceed the threshold, with angles between 150 and 180 degrees adjusted similarly. The resampled coordinates, marked with red x symbols in Fig. 5., are immediately updated, and their new positions are used to calculate the angles of subsequent coordinates. This continuous updating creates a chain reaction of adjustments, resulting in a set of coordinates that can be further smoothed using curve-fitting techniques. The impact of the downsampling rate and resampling process on the path's appearance is detailed in Section VI.

Finally, several curve-fitting, including Dubins, Bezier, polynomial, Reeds-Shepp, cubic, and B-Spline curves, were evaluated. The results indicate that all methods are suitable for path approximation, as the resampling process resolves sharp corner issues. Although polynomial curves were used in the final implementation, similar results could be achieved with other methods. The smoothed path is shown as a blue continuous curve in Fig. 5.

VI. RESULTS AND DISCUSSION

The hypothesis presented in this paper is that the vessel path planning and interpolation methods described are effective for real vessels of medium and smaller lengths and higher turning rates. The path planning algorithm aims to guide the vessel along a feasible and safe route. Optimization criteria include physical feasibility, ensuring smooth paths with gentle turns, and movement uniformity, which manages speed changes using a safe zone, as detailed later in the section. The criteria assume the path is planned at a safe distance from the coast, allowing vessels to exceed speed limits in designated zones to minimize travel time. Safe navigation and reduced travel time are related, as vessels can travel faster in areas over 300 meters from the coast, improving response times to unpredictable factors. To achieve this, the path is modeled using costs based on distance from the coast and adjusted interpolation parameters.

One use-case uses a region of a peninsula near the Krk Bridge in Croatia, as shown in Fig. 6. and 7. The route crosses a marine channel and navigates around prominent capes and a strait. The path interpolation process can obscure zone boundaries, leading to unwanted transitions between distance zones. This can result in abrupt speed changes, which are harmful to the vessel. Fig. 6. highlights these issues, showing that paths often traverse multiple zones due to the interpolation process,

To mitigate this, a safe zone is established 300 to 350 meters from the coast, highlighted in brighter green in Fig. 6. A safe zone helps avoid problematic transitions and shifts the path further from the 300-meter limit as its cost value is greater than 1. The figure illustrates how this adjustment resolves the issues related to zone transitions. Proper cost settings for different zones are crucial: green zone should have slightly higher costs than safe zone, the cost difference between yellow and green zones should be significantly greater than the difference between safe and green zones, and red zone should have the highest costs to discourage passage through hazardous areas. This cost ratio ensures that paths favor safer and more feasible routes.

The final factor to consider is the impact of the downsampling rate on the path appearance. As shown in Fig. 8, using downsampling rates greater than 10 results in poor approximation of the D* Lite path, which can introduce safety risks. However, higher downsampling rates produce smoother paths with a larger turning radius. Adjusting the path curvature via the downsampling parameter ensures that the algorithm can be adapted for various vessel types with different turning rates.

To validate the conclusions on cost values and downsampling parameters, a complex test area around Cape Šilo and Soline Bay was used, as shown in Fig. 9. The narrow strait in Soline Bay posed an additional challenge for the D* Lite cost-planned path and coordinate resampling concept due to its width of less than 300 meters, where green and safe zones are absent. As expected, the D* Lite path (magenta) was less smooth and non-feasible. The interpolated path (cyan) met the optimization criteria, but in the strait, the path came too close to the red zone, raising a safety concern. This issue cannot be resolved simply by adjusting the downsampling parameter or cost map ratios, as each zone's cost is constant. One possible approach to address this issue is to introduce a monotonically decreasing cost for each zone from the coastline outward.

This example demonstrates that, even without a dynamic vessel model, it is possible to implement a functional path planning algorithm using the D* Lite algorithm and interpo-

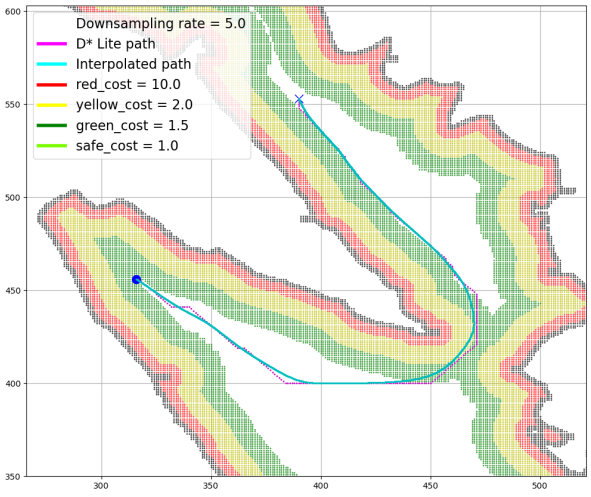


Fig. 6: Path results safe zone cost 1

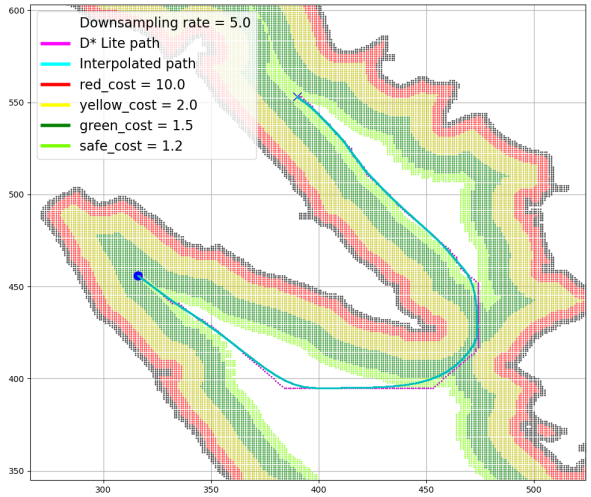


Fig. 7: Path results safe zone cost greater than 1

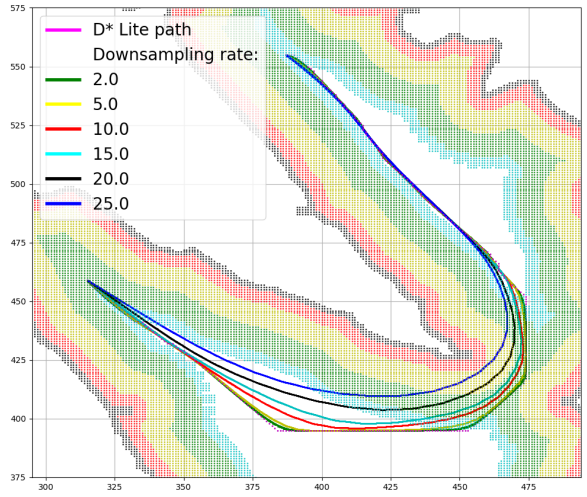


Fig. 8: Impact of the Downampling Parameter on Path Appearance.

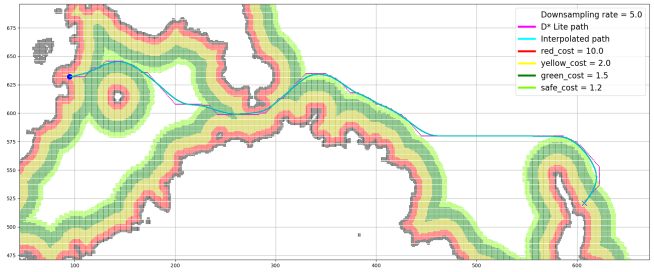


Fig. 9: Results of the path planning tests in the area of Cape Šilo and Soline Bay.

lation techniques. By detecting and resampling sudden turns in the D* Lite raw path, a feasible and smooth path can be obtained. Selecting the appropriate downsampling rate parameter allows direct control over the path's curvature, enabling the turning rates to be adapted for different types of vessels. Although there are challenges that need to be overcome for this system to be fully applicable in real-world vessel path planning, the system architecture and performance, with the proper upgrades, have the potential to become part of real USV systems in the future.

VII. CONCLUSION AND FUTURE WORK

This paper presented a global vessel path planning system architecture based on the ROS2 framework. The system components and communication topology were introduced, along with the proposed path planning algorithm. The algorithm generated smooth, feasible paths for small and medium-sized vessels using a cost map based on OpenStreetMap data. Adaptable path curvature was achieved through the application of the D* Lite algorithm and interpolation techniques, ensuring safe navigation by avoiding abrupt transitions. Simulations conducted using real-world coastal maps of the Croatian Adriatic demonstrated the system's potential for integration into real-world USV systems, providing reliable and efficient path planning solutions.

Future work will focus on enhancing the cost map by introducing non-uniform zone costs and incorporating hydrographic and safety features. Path planning using a dynamic vessel model will be compared to the presented approach, and the issue of paths crossing obstacles after coordinate resampling will be addressed. The system will be extended to support local real-time replanning with COLREG rules to avoid collisions with previously unknown static and dynamic obstacles. This will enable the vessel to converge back to the preplanned global path.

ACKNOWLEDGMENT

This paper is a summary of the publicly available thesis, *Model-Informed Path Planning and Control of Autonomous Vessels* (2024), developed at [2].

The authors acknowledge the use of ChatGPT, an AI language model by OpenAI, for assistance with grammar checks and context clarification. The AI was not involved in original research, analysis, or content creation but helped improve the paper's clarity and readability.

REFERENCES

[1] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abm6074

[2] D. L. A. M. M. F. S. Macenski, T. Moore, "From the desks of ros maintainers: A survey of modern capable mobile robotics algorithms in the robot operating system 2," *Robotics and Autonomous Systems*, 2023.

[3] C. Seale, T. Redfern, P. Chatfield, C. Luo, and K. Dempsey, "Coastline detection in satellite imagery: A deep learning approach on new benchmark data," *Remote Sensing of Environment*, vol. 278, p. 113044, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0034425722001584

[4] J.-A. Meyer and D. Filliat, "Map-based navigation in mobile robots:: Ii. a review of map-learning and path-planning strategies," *Cognitive Systems Research*, vol. 4, no. 4, pp. 283–317, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138904170300007X

[5] Y. Wu, T. Wang, and S. Liu, "A review of path planning methods for marine autonomous surface vehicles," *Journal of Marine Science and Engineering*, vol. 12, p. 833, 05 2024.

[6] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org , 2017.

[7] H. Yuan, "Method and system for performing services in server and client of client/server architecture," Patent, 05, 2014.

[8] S. Battersby, M. Finn, E. Usery, and K. Yamamoto, "Implications of web mercator and its use in online mapping," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 49, pp. 85–101, 06 2014.

[9] E. Winarno, W. Hadikurniawati, and R. N. Rosso, "Location based service for presence system using haversine method," in *2017 International Conference on Innovative and Creative Information Technology (ICITech)*, 2017, pp. 1–4.

[10] T. Ministry of Maritime Affairs and Infrastructures, "Act on amendments of maritime code, official gazette, no. 61/2011 and other related amendments," Zagreb, Croatia, 2013, pursuant to Article 64, paragraph 1, point 3 of the Act on Amendments of Maritime Code and Article 24, paragraph 1, and in connection with Articles 64a and 75a.

[11] A. Sakai, D. Ingram, J. Dinius, K. Chawla, A. Raffin, and A. Paques, "Pythonrobotics: a python code collection of robotics algorithms," 2018.

[12] S. Koenig and M. Likhachev, "D* lite," in *Proceedings of the National Conference on Artificial Intelligence*, 01 2002, pp. 476–483.