Winter 2018

Providing Nautical Chart Awareness to Autonomous Surface Vehicles

Samuel John Reed
University of New Hampshire, Durham

PROVIDING NAUTICAL CHART AWARENESS TO AUTONOMOUS SURFACE VEHICLES

BY

SAMUEL REED

BS, University of New Hampshire, 2015

THESIS

Submitted to the University of New Hampshire

In Partial Fulfillment of the Requirement for the Degree of

Master of Science

In

Electrical Engineering

December, 2018

This thesis has been examined and approved in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering by:

Thesis Director, Dr. Brian Calder,

Center for Coastal & Ocean Mapping Associate Director

Affiliate Professor of Electrical and Computer Engineering

Val Schmidt,

Research Project Engineer, Center for Costal and Ocean Mapping

Dr. Se Young Yoon

Assistant Professor of Electrical and Computer Engineering

Dr. Kent Chamberlin,

Department Chair and Professor of Electrical and Computer Engineering

on 11/27/2018.

Approval signatures are on file with the University of New Hampshire Graduate School.

ACKNOWLEDGEMENTS

Sam Reed would like to thank many people including Val Schmidt who was vital in guiding Sam through his research and field work as well as the rest of the robotics lab at the Center for Coastal and Ocean Mapping at UNH. Additionally, Sam would thank the UNH Electrical Engineering department in help supporting his research. Sam would also like to thank his family and Liz Weidner for their encouragement and support over the last three and a half years. Finally, Sam would like to thank NOAA as this work was funded through the NOAA Grant NA15NOS4000200.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

ALFUS – Autonomy Levels for Unmanned Systems

ASV – Autonomous Surface vehicle

AIS – Automatic Identification System

ARPA – Automatic Radar Plotting Aid

CoTS – Commercial off the Shelf

DEM – Digital Elevation Model

ECDIS – Electronic Chart Display and Information System

ENC – Electronic Nautical Chart

FAA* – Finite Angle A*

GDAL – Geospatial Data Abstraction Library

GPS – Global Positioning Satellite

GUI – Graphical User Interface

IHO – International Hydrographic Office

IvP – Interval Programming

LPF – Low-Pass Filter

MLLW – Mean Lower Low Water

MHW – Mean High Water

MOOS – Mission Oriented Operating System

NOAA – National Oceanography and Atmospheric Administration

RNC – Raster Nautical Chart

RTK – Real Time Kinematic

UNH – University of New Hampshire

UTM – Universal Transverse Mercator

VALSOU – Value of Sounding Attribute

WATLEV – Water Level Effect Attribute

WGS84 – World Geodetic System 1984

ABSTRACT

PROVIDING NAUTICAL CHART AWARENESS FOR AUTONOMOUS SURFACE VEHICLES

by

Samuel Reed

University of New Hampshire, December 2018

Autonomous surface vessels (ASVs) have many applications in both military and civilian domains including mine countermeasure, seafloor mapping, and physical oceanography. However, to act as effective tools, ASVs require high levels of autonomy. Currently, many commercially available ASVs have static mission plans with minimal awareness of their environment, which results in a labor intensive approach that does not scale to management of multiple vehicles. In this research, ASV autonomy was increased through the development of an intelligent mission planner and a real-time obstacle avoidance system utilizing Electronic Nautical Charts (ENCs), which describe known hazards in the marine environment without suffering from the challenges of real-time sensor processing. A new algorithm called Depth-Based A* was developed as the mission planner, where the nominal A* search algorithm was expanded by utilizing a novel cost function that balances driving in the channel with taking the most direct route on an ENC-derived cost map. Although charted obstacles can typically be avoided through mission planners, there is still an advantage in having the code do this. However, since it enables even higher levels of autonomy (e.g., “go in this area, but avoid all known obstacles”) they must still be accounted for in real time as other behaviors (i.e., avoiding uncharted obstacles or vessels) might cause the ASV

to deviate from the planned path. The reactive obstacle avoidance system developed in this research reorganizes the ENC into a quick-search database where ENC-based obstacles in the ASV's proximity are determined and avoided. These algorithms were tested with both a Seafloor System EchoBoat and ASV Global C-Worker 4 in simulation and in the field using an EchoBoat, where they avoided both concave and convex polygons. The algorithms developed in this research provide the ASV with a higher level of autonomy, potentially allowing for the same number of human operators to manage more ASVs.

CHAPTER 1

INTRODUCTION

Autonomous surface vehicles (ASVs) have enormous potential as they have the ability to replace one manned vessel with multiple ASVs for the same number of operators. This replacement reduces the danger to operators and cost of operation for defense, industrial, and scientific applications. ASVs are defined as unmanned vessels that navigate on the water's surface independent of direct human control. In the military domain, potential applications include harbor security [1], mine countermeasures [1], anti-submarine warfare [1] and perimeter patrol [1], [2]. Civilian applications include commercial shipping, seafloor mapping [3],[4],[5], pollution control [6] and physical oceanography [7]. ASVs can range from small and highly maneuverable vessels suitable for shallow coastal areas and rivers to large vessels with long endurances.

For an ASV to be an effective tool, a high level of autonomy is necessary as this decreases the amount of human oversight and as a result, increases scalability. An ASV must be able to fulfill its desired mission and adjust its planned path to avoid potential hazards without direct human oversight and intervention. Additionally, the ASV must be able to interpret its internal states and understand if it can complete its current mission.

Many of the currently commercially available vessels lack the necessary autonomy to reach their potential. Most vessels can autonomously navigate in a straight line between waypoints by maintaining a desired speed and heading, but they often lack knowledge of the environment around them and algorithms to safely navigate through it; human operator intervention is required for

these ASVs to avoid obstacles. This environmental blindness and the lack of autonomy results in labor-intensive mission planning and operations, severely limiting the feasibility of ASVs.

Using the Autonomy Levels for Unmanned Systems (ALFUS) scale, defined in [8] and shown in Table 1.1, the present state of many commercial off the shelf (CoTS) vessels would be defined at ALFUS Level 3 – capable of executing pre-planned missions, but no knowledge of the local or planned environment (Level 4), no ability to detect and avoid hazards (Level 5/6), and little in the way of higher level capabilities.

To unlock their potential, ASVs must utilize a hybrid mission planning and autonomy system consisting of an *a priori* mission planner and a real-time, reactive obstacle avoidance system, where both systems are cognizant of charted hazards to navigation. We feel that vehicles cannot unconditionally follow a pre-programmed list of waypoints nor react to other vessels in real time without regard to these hazards.

When human mariners navigate into unknown ports, they would first consult a nautical chart as it provides a holistic view for the mariner to plan safe passage. They would use the chart

	Autonomy Level
1	Remote Control
2	Remote Control with Vehicle State Knowledge
3	External Pre-Planned missions
4	Knowledge of local and planned path environment
5	Hazard avoidance or negotiation
6	Object detection, recognition, avoidance, or negotiation
7	Fusion of local sensors and data
8	Cooperate operations
9	Collaborate operations
10	Full Autonomy

Table 1.1: Autonomy Levels for Unmanned Systems (AFLUS) Scale [8]

to plan a path free of obstacles in deep water to ensure the vessel would not run aground. Invariably, when navigating the path, they would have to react to other vessels or obstacles that were not on the chart, and have to alter their course. When they do so, they will consult the chart again to ensure that their new course does not put them in danger. An ASV should operate similarly, planning missions with the knowledge of what is already known and be able to react safely in real-time to changing conditions. Thus, this research is devoted to increasing the autonomy of ASVs by providing situational awareness through both a mission planner and a reactive obstacle avoidance system utilizing Electronic Nautical Charts (ENCs).

1.1: Electronic Nautical Charts (ENCs)

Nautical charts give context to a mariner's vessel location by providing the mariner with an indication of his environmental setting, such as the locations of hazardous objects, depth for underwater objects, contours and soundings, classification of lights, buoys and beacons, and descriptions of topographic landmarks. Nautical charts are a vital component to human maritime navigation as they allow the navigator to integrate the ship's position, speed, and heading along with the information on its environment to them help plot a safe course.

Traditionally printed on paper, nautical charts are a cartographic representation of objects optimized for the use of mariners. In recent years, nautical charts have been converted into digital form. There are two types of digital charts: Raster Nautical Charts (RNC) and Electronic Nautical Charts (ENCs). RNCs are a digitized raster image of the paper nautical chart.[1] This raster cartographic representation makes machine interpretation extremely difficult. On the other hand, ENCs are produced in a vector format where each object type is semantically separated into

[1] Originally, United States' RNCs were produced by digitizing the paper charts, however they are currently made digitally and are only printed on demand.

different layers, and are encoded using the International Hydrographic Office (IHO) S-57 Standard [9]. A representation of a section of an ENC is shown in Figure 1.1. ENCs have significant advantages over RNCs: the semantic meaning of objects and metadata is easily understood in ENCs, they have significantly smaller file sizes, and individual elements are easily accessible. In the United States, nautical charts, in both raster and ENC form, are produced by the National Oceanic and Atmospheric Administration (NOAA) and are readily available on NOAA's website (https://nauticalcharts.noaa.gov/).[2]

Both RNCs and ENCs have scales associated with their data and display. RNCs have a static scale which matches the scale from the corresponding paper chart. For example, the scale of

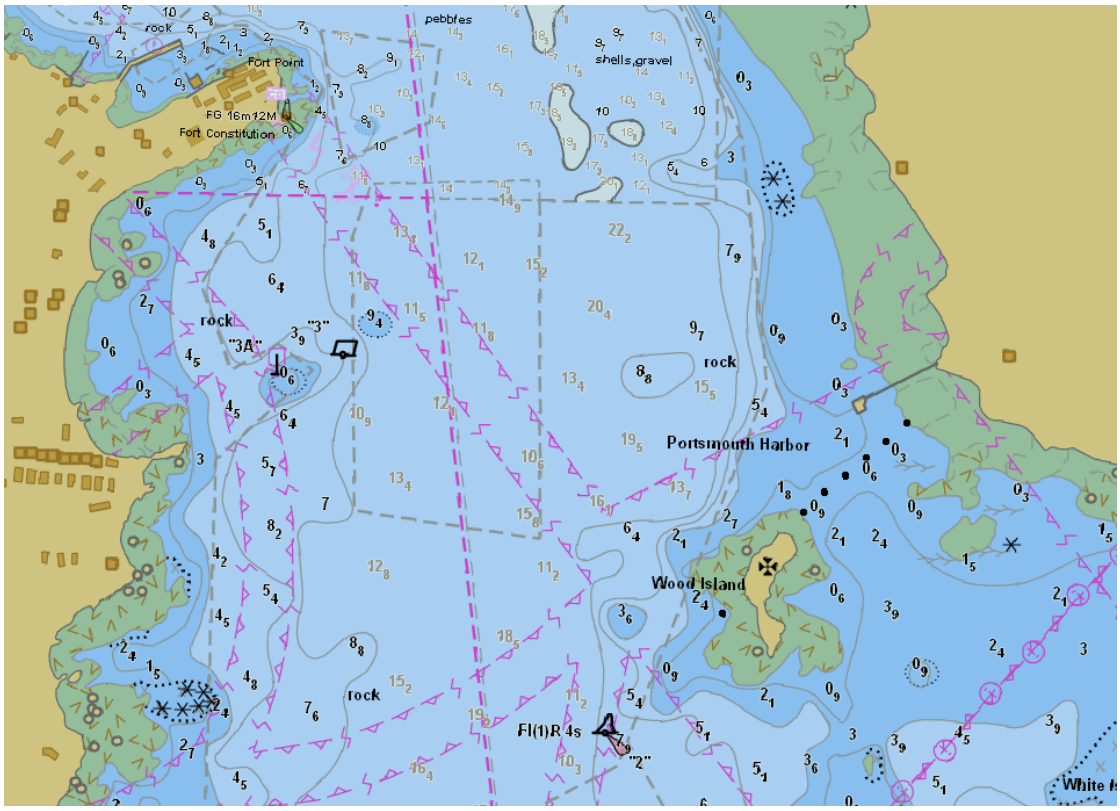


Figure 1.1: Section of an Electronic Nautical Chart (ENC) in the Portsmouth Harbor (US5NH02M) showing the University of New Hampshire's (UNH's) Pier in the top left corner.

[2] In other countries ENCs are not free, but are still available at a cost.

a typical harbor chart is 1:20,000, where one millimeter on the chart corresponds to 20 m in real life. As a result, it is prudent (and recommended) to use the largest scale available chart when navigating. ENCs are vector charts and encoded into each ENC is a scale for display, including all subsets, which match the corresponding RNC.

ENCs are composed of different layers, organized by object type (e.g. underwater rock, wreck, or sounding). Within each layer, all the individual objects in are stored as "features". Each individual feature has many attributes that describe the object. For example, attributes of underwater rocks include quantitative and qualitative depth information (e.g. "5 meters" or "covers and uncovers during the tidal cycle"), quality of the sounding measurement, and the name of the rock. As ENCs hold pertinent information on the environment for vessel safety without needing additional sensors, ENCs are used as the data source for all algorithms in this research.

The combination of shoal biasing during the gridding and compilation of charts as well as the chart being referenced to Mean Lower Low Water (MLLW)[3] provides mariners with the assurance of vessel clearance largely without regard to tides. However, many ASVs operating in coastal environments are small, man-deployable crafts having small drafts and therefore navigation relative to MLLW can be overly conservative, particularly when the tidal range is large (as tides vary globally from negligible to over 10 m). Consequently, it is more practical to account for the current tide, so that the ASV does not make unnecessarily conservative course corrections.

[3] Nautical charts produced by NOAA are referenced to Mean Lower Low Water (MLLW) for depths and Mean High Water (MHW) for drying heights. MLLW is the average of the lower of the two daily tides over a 19-year period and MHW is the average height of the high tides over a 19-year period. MLLW is used for nautical charts to ensure that the measured depth of the water is unlikely to be lower than the charted depth.

In this research, charted depths are adjusted in real time to prevent an overly conservative estimate of grounding risk when the tidal range greatly exceeds the vessels draft.

1.2: Previous Work

Considerable work has been done on increasing the autonomy of ASVs though intelligent path planning and reactive obstacle avoidance. *A priori* maps of the marine environment, such as satellite images and OpenStreetMaps of coastal areas as well as nautical charts, have been used by others in their planning methods. Path planning can be classified into three categories: *a priori* mission planning, reactive obstacle avoidance, or a hybrid approach.

Yang [10] defined a new search algorithm called Finite Angle A* (FAA*) to plan paths from satellite images. FAA* is a graph search algorithm that modifies the classic A* algorithm [11] (pronounced A-Star) by limiting the heading changes between waypoints. FAA* differed from the classic A* implementation by increasing the search area from the eight Moore Neighbors to 180 nodes through increasing the branching factor from one to eight. This increase in branching allowed Yang to disallow sharp turns by setting a maximum angular displacement between waypoints and prevents the creation of paths with turns sharper than can be practically navigated.

Naus and Waz [12] used the A* graph search algorithm on a nautical chart as a mission planner for ASVs. Naus and Waz utilized the depth area and other features that define obstacles to translate the ENC to a binary graph used by A*.

Sauze [13] developed a reactive mission planner for a robotic sailboat using a trajectory ray tracing method on an OpenStreetMap. Sauze's algorithm casts 150-meter rays around the desired heading. If the ray intercepts land, then the algorithm selects the closest path to the desired

with a five-degree buffer. One of the limitations of Sauze's work was that the algorithm could get stuck in local minima in which no satisfactory path could be found.

Larson [14], [15] used a hybrid approach where obstacle avoidance was split into a far-field and a near-field reactive planner. In the far-field, 2D obstacle maps were created with moving contacts from Automatic Identification System (AIS) and Automatic Radar Plotting Aid (ARPA) as well as stationary contacts from nautical charts. Obstacles from nautical charts were classified into four categories: above water, on the water surface, below water, and land. To find the path through the environment described with the 2D obstacle map, Larson used the A* graph search algorithm, which included a cost for the ASV being in close proximity to the obstacles.

In Larson's reactive planner, data was fused from obstacle avoidance sensors and the 2D obstacle map to create a world model. In Larson's approach, arcs, whose radius is a function of velocity and turn rate, are projected in front of the ASV over the local world-model obstacle map and decisions are made by a weighted voting system using votes from the fused sensors, a free-space behavior, and the path planner.

Casalino [16] used a hybrid approach to obstacle avoidance consisting of three layers: static, moving, and short term. In the static layer, visibility graphs determined collision free paths through the environment known *a priori* through a chart server. Then, Dijkstra's Algorithm [17] was used to determine the optimal path. Dijkstra's Algorithm is a graph search algorithm utilizing a flood fill approach. In the moving layer, bounding boxes were placed around moving obstacles to account for their potential movements and A* was used to determine the optimal, collision-free path. The final layer was not discussed in the paper, however its goal is to provide a safety fallback if the first two layers should fail.

1.3: Contributions of this Thesis

For ASVs to reach their potential, their autonomy must be increased from unquestionably following waypoint-based mission plans derived by human-operators to utilizing intelligent mission plans that autonomously avoid known obstacles. In this thesis, a safe, environmentally aware mission plan was generated with a variant of the classic A* [11] search algorithm, similar to the approaches taken in [10], [12], [14] and [15]. In this A* variant, the method of increasing the branching factor was adopted from [10] to allow for a smoother path. Additionally, a novel cost function that incorporates a depth-based cost was developed, which allows for prioritizing staying in deep water, like a human mariner. To incorporate a depth based-cost in the mission planner, the graph used in path planning for this thesis was created by interpolating and generalizing an ENC into a gridded cost map, where the cost in each cell is a proxy for risk. This graph, although utilizing ENCs like the approaches in [12], [14], [15], and [16], differs from ones used in the previously discussed works as it has depth values for each cell in the graph. Further details of the mission planner are described in Section 2.1.

As the maritime environment is not static, it is also necessary for the ASV be able to avoid uncharted obstacles, like other vessels, and when doing so, maintain safety around charted obstacles in real-time using a reactive obstacle avoidance system. In this system, charted obstacles are categorized by the threat they pose to the ASV, similar to [14] and [15]. However, in this thesis, the proposed threat level attribute has a finer gradation, which allows the reactive obstacle avoidance system more degrees of freedom for its avoidance strategies. Next, all headings that lead toward obstacles are determined through an angular sweep method similar to the ray casting method in [13]. Once these headings have been determined, they are penalized using MOOS-IvP [18], [19], [20]. As a result, this thesis implements its chart-based reactive obstacle avoidance

system in a behavior based infrastructure, similar to [14] and [15]. However, in this thesis, obstacles in a full 360-degree sector are considered to ensure the safety of the ASV. Finally, tidal predictions are incorporated into the reactive obstacle avoidance system of this thesis unlike the other previously mentioned works ([10], [12]-[16]) as tidal predictions allows for more accurate risk assessment and helps minimize unnecessary avoidance procedures. Details of the real-time obstacle avoidance system are described further in Section 2.2.

The research developed in this thesis helps increase autonomy of CoTS ASVs, from ALFUS level 3 to between levels 4-5 by providing an environmentally aware mission plan and guidance to the helm when its intended path is unsafe due to known hazards to navigation. This increase in autonomy was implemented using an A*-based mission planner on top of a nautical chart-derived cost map to provide the ASV a safe path plan and a real-time object avoidance system to avoid charted hazards during unplanned maneuvers.

Chapter 2

Method

Section 2.1: ENC Derived Mission Planner

Section 2.1.1: A* Search Algorithm

Vessel path planning is an integral piece of safe decision making for ASVs. Since the inception of robotics, solving the problem of safe path planning has been the objective of many research efforts. One of the forefathers of path planning research, Edsger Dijkstra, published "A Note on Two Problems in Connexion with Graphs" [17] in which he proposed an algorithm for finding the shortest path between nodes, known colloquially as Dijkstra's Algorithm. Dijkstra's Algorithm determines the optimal path from the source node to all other potential nodes in an exhaustive search. As a result, this algorithm proved inefficient and in Peter Hart's paper [11], he improved the performance of Dijkstra's Algorithm through a "best-first search" method called A*. A* utilizes a heuristic function to guide exploration by preferentially searching nodes closer to the goal. This research utilizes the A* search algorithm for path planning purposes with several modifications adopted from others to better accommodate marine path planning. The implementation is described Section 2.1.2.

The A* search algorithm works within a spatial domain known as a "graph" which may be a binary grid [10], [12], probabilistic occupancy grid [14], [15], visibility graph [16] or others. The graph is simply a representation of possible states with some cost associated with a transition between any two adjacent states. The graph in this thesis utilizes a weighted cost map in the form

a rectilinear grid, which is defined by gridding the topographical features from an ENC at a resolution of 0.25 mm of the ENC's compilation scale. For example, a chart with a compilation scale of 1:20,000 is gridded at a scale of 5 m. This resolution strikes a balance between over interpolating sparse data having sufficient granularity for navigation, and computational cost. However, it is recommended to use the largest scale chart available when navigating using ENCs as they will produce grids with the greatest spatial resolution and have the least number of generalizations. A detailed description of the grid creation is given in Section 2.1.3.

Safe navigation within A* path planning modules also requires consideration of the size of the vessel as large vessels are less maneuverable and more prone to grounding at a given water depth. For path planning in this thesis, prohibited areas and obstacles to be avoided are classified as any topographic feature with a depth less than three times the ASV's draft. Additionally, the vessel's dimensions are used elsewhere (Section 2.2) as weighting factors in its response to obstacles in real-time.

Given a starting point and a desired destination on a gridded map, A* finds the optimal route between the points, for the given angular resolution. On the first iteration, cells in the immediate vicinity of the starting point are considered as candidate paths, prohibited cells are omitted, and the remaining cells create a "frontier" between explored and unexplored areas. The total projected cost of traversing through each cell in the frontier to the goal is calculated and the cell with the lowest cost is removed from the frontier having been explored; expansion of the frontier is begun anew from this cell. Finally, the explored cell is annotated with the coordinates of its prior cell, incrementally building a candidate path back through the graph. On successive iterations, the frontier is expanded, new cells are evaluated, and those with the lowest projected

cost to reach the goal are explored preferentially. When the goal point has been reached, the identified path through the map is the succession of least-cost cells pointing back to the start node.

During each iteration of exploration as described above, A* determines which cells in its frontier have the lowest total projected cost traveling from the start node through the current node to the goal node. The total projected cost is calculated using the cost function:

$$F(n)=G(n)+H(n) \tag{2.1}$$

where F(n) is the total projected cost, G(n) is the cumulative cost to reach the current node, and H(n) is the estimated cost from the current node to the goal node, also known as the heuristic. Once the total projected cost for a node has been calculated, the node is considered "evaluated". In the classic implementation of A*, this cost function minimizes the total path length by only considering the distance traveled and the projected distance to go for G(n) and H(n) respectively.

Classic A* implementation adds the eight nearest neighbors (Moore Neighbors) to the frontier during each iteration. The Moore Neighbor search is defined as having a branching factor of one and is shown in Figure 2.1. This procedure limits the heading changes to one of eight possible directions, which often results in an irregular and/or impossible path for shipboard

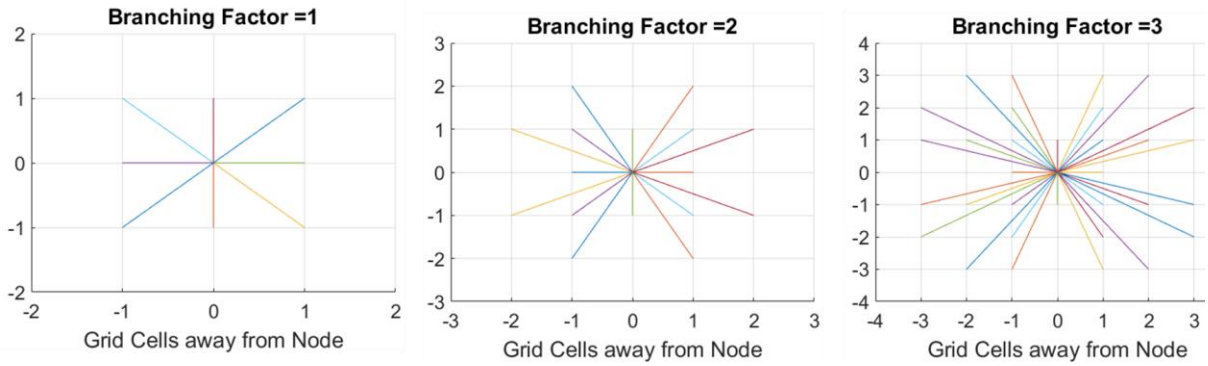


Figure 2.1: The plots here indicates the cells considered at each exploration step in A* when various branching factors are used. Increasing the branching factor allows for finer resolution of heading changes and more realistic path generation for marine vehicles.

navigation. In this research, the number of neighbors searched was expanded to 180 (branching factor of eight) using the method in [10], therefore allowing heading changes of less than two degrees in each step of the path. To illustrate the expanded branching factors, Figure 2.1 shows factors of one, two and three, searching eight, sixteen and thirty-two nodes respectively (a branching factor of eight is not easily depicted). Expansion is deemed valid for non-adjacent cells if the straight-line path between the two nodes does not pass through prohibited cells (i.e. obstacles).

Once the cost for all frontier nodes have been evaluated, the node with the lowest total projected cost is used to expand the frontier through the branching process. The key for minimizing exploration in A* is the heuristic function, defined in this work as the Euclidean distance from the current node to the goal node. By including the heuristic in the total cost and by choosing the node with the lowest total cost for expansion, A* minimizes unnecessary exploration.

An example of A* with a branching factor of 2, cost, G(n), based on minimum path length, and a Euclidean-distance-based heuristic, H(n), is shown in shown in Figure 2.2. The graph is represented by a 4x4 grid in which the centers of each box represent possible vehicle locations. Obstacles (squares for which passage is forbidden) are represented in blue and the cell that is being expanded is denoted with the yellow square. In this example, the start position is node C1 and the goal is node D4. On the first iteration, the neighbors that can be reached from C1 with a branching factor of two are identified (nodes D1, B1, A2, B2, C2, D2, B3, and D3, marked in purple). Although nodes B1, A2, D2, and D3 are potential neighbors to C1, they were omitted from consideration during expansion as the path from node C1 to the candidate node would travel through an obstacle. Therefore, only nodes D1, C2, B2 and B3 are considered and each of these nodes are placed in the frontier. The cumulative cost to get to each square, *G*, the heuristic

(calculated as the Euclidean distance to the goal), H, and the total projected cost of passing through the cell ($F = G + H$), are calculated and shown for each cell in the frontier. After the first expansion, the node with the smallest total cost in the frontier, in this case C2, is subsequently expanded.

On the second expansion, the neighbors of C2 that can be reached with a branching factor of two are again identified (A1, D1, B2, D2, A3, B3, C3, D3, B4 and D4, outlined in purple). Nodes A1 and D2 still cannot be explored as the path to reach these nodes would traverse through obstacles, but all other nodes are placed in the frontier. The cumulative cost, heuristic, and total projected cost are again calculated for the remaining nodes. However, since the cumulative cost to go from node C2 to nodes D1, B2 and B3 is higher than the previously stored cost for those cells, those values are not updated. This process ensures that the optimal path, given a branching factor of 2, is always taken during path reconstruction. After this second expansion, the node with the

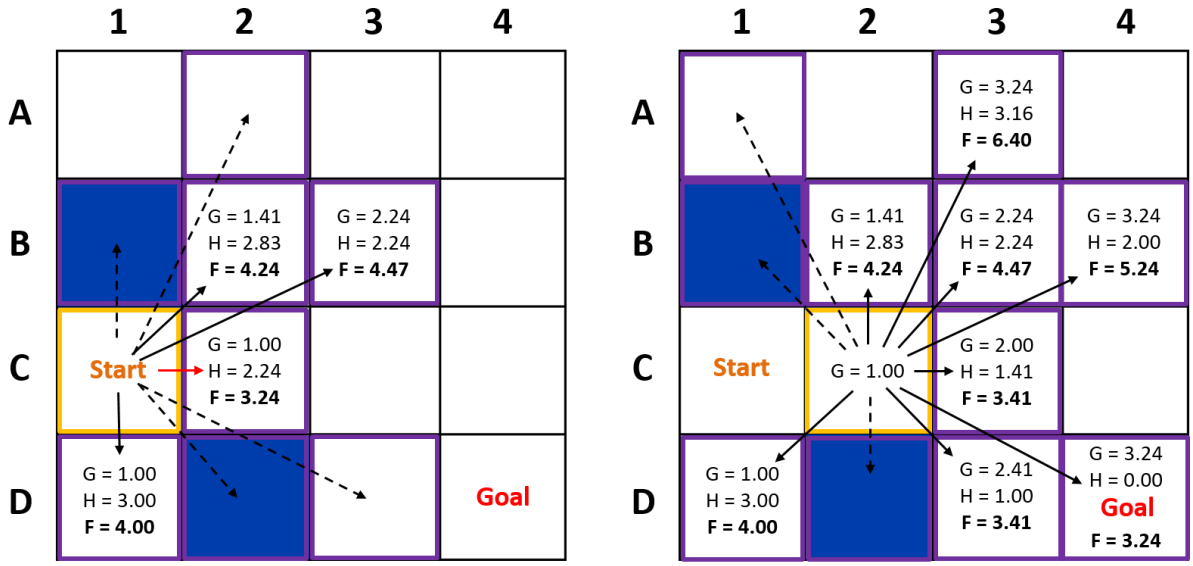


Figure 2.2: Example of the A* algorithm with a branching factor of 2 and the heuristic as the Euclidean Distance, where the node being expanded is outlined in yellow and the neighbors that can be reached during expansion in purple. The left image shows the first expansion (C1) and the right image shows the second expansion (C2).

lowest cost in the frontier is D4. Since D4 is the goal node, the algorithm terminates and the path is C1 -> C2 -> D4.

Section 2.1.2: Building the Graph from ENCs

In this research, the graph for the A*-based mission planner was an interpolated, depth-based grid created from the vector features in ENCs that incorporate depth information including soundings, rocks, wrecks, land areas, buoys, etc. First, the vector features were converted from World Geodetic System 1984 (WGS84) to Universal Transverse Mercator (UTM) and were buffered using the widely used Geospatial Data Abstraction Library (GDAL) [21]. In this research, the buffer (B) took into account the vessel length (L), width (W), and a user defined safety margin (SM) (Equation 2.2). The buffer accounts for the potential positioning and control errors as well as allowing A* to treat the ASV as a point object as the ASV's dimensions are now accounted for by buffering all obstacles. In this work, the Safety Margin is defined as two, but could be modified in future work to account for uncertainty in the underlying data or ship characteristics.

$$B=\frac{SM}{2}*\sqrt{L^2+W^2} \tag{2.2}$$

In this work all points were also buffered at two millimeters (diameter) at chart scale as many objects that are represented as points in charts are not actually points and there is no official uncertainty measurement for these objects. This buffer may not be appropriate for all chart and vessel sizes and should be further explored in future work. However, in this work buffering of objects with point-based geometry was accomplished by replacing the original point with a diamond-shaped mask of points whose spacing is at the grid resolution and diameter is 2 mm at chart scale.

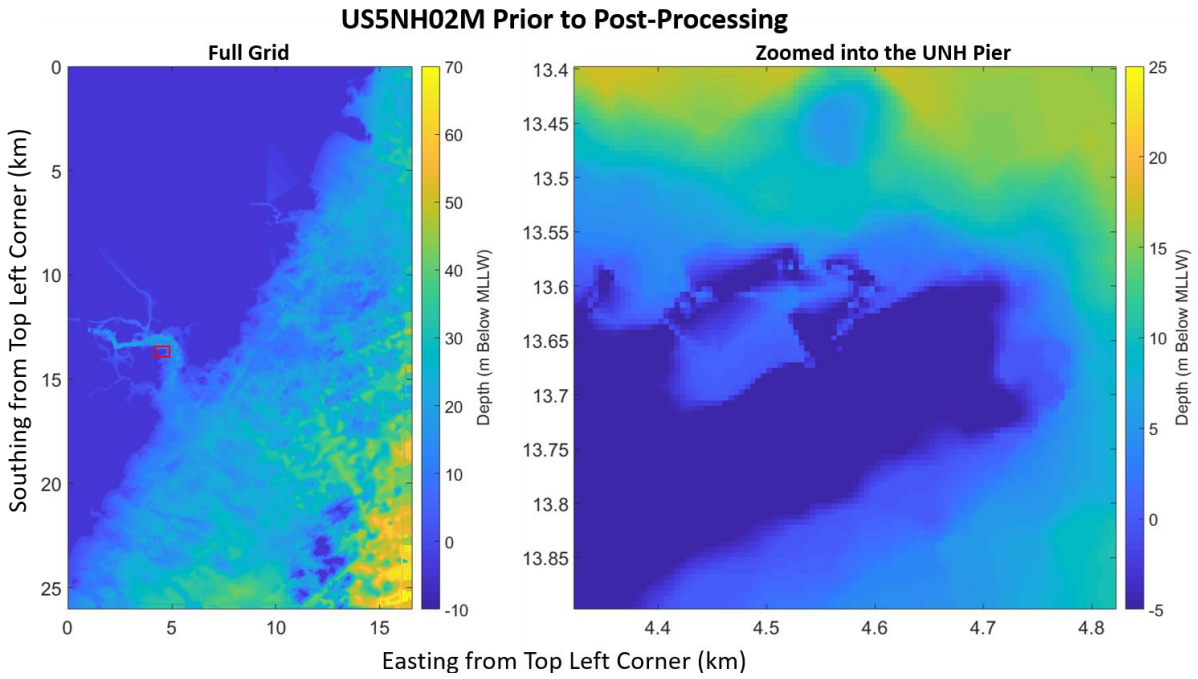


Figure 2.3: The interpolated grid for the Portsmouth, NH ENC (US5NH02M) before post-processing. The left image gives the full grid and the right zooms into the area around the UNH Pier. The red box in the left image defines the bounds of the area shown in the right image.

Next, the vertices of the polygons derived from these features and the diamond-shaped point masses were linearly interpolated to the grid's resolution using GDAL. This interpolation produces an error-ridden surface (as shown in Figure 2.3, where the left image gives the full grid for the Portsmouth Harbor ENC and the right zooms into the area around the UNH Pier) as polygons are only represented by their outline and when buffered, they may intersect causing interpolation artifacts. Therefore, after interpolation, a conflict resolution process is needed.

Additionally, when there is more than one depth measurement inside a grid cell, it is possible that a linear interpolation will produce an estimated depth that is deeper than the shoalest depth measurement. Were this to happen, the resulting graph may prove unsafe for navigation by failing to conservatively represent the depths of hazards. This has been mitigated through the conflict resolution process described below.

To begin the conflict resolution process, a series of rasterized masks of identical size to the interpolated grid were created. The M_COVR raster, created by an ENC feature of the same name, was generated to delineate the outer edges of the chart. The DA_shoal (Figure 2.4a) and DA_deep rasters (Figure 2.4b) were created to enforce the vertical bounds of each depth area after

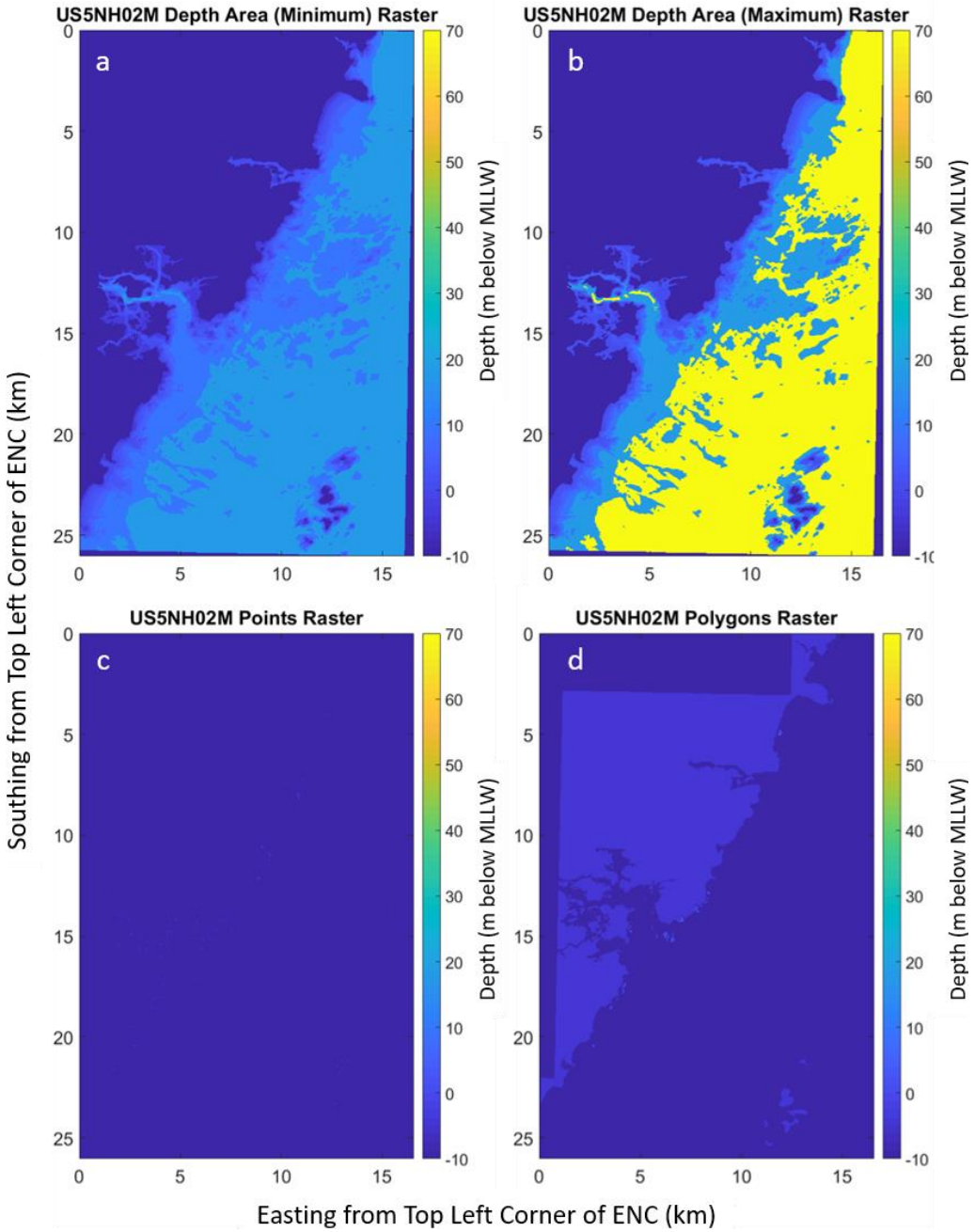


Figure 2.4: Images showing the rasters used in the conflict resolution process where the top images show DA_shoal (2.4a) and DA_deep rasters (2.4b) and the bottom images show the Point (2.4c) and Polygon Rasters (2.4d).

Water Level	General Description	Depth Range	
		Atlantic	***Pacific***
1	Partly Submerged at High Water	---	---
2	Always Dry	1+ feet above MHW	2+ feet above MHW
3	Always Underwater/ Submergered	1+ feet below MLLW	2+ feet below MLLW
4	Covers and Uncovers	1 foot below MLLW to 1 foot above MHW	2 feet below MLLW to 2 foot above MHW
5	Awash	1 foot below MLLW to 1 foot above MLLW	2 feet below MLLW to 2 feet above MLLW
6	Subject to Indundation or Flooding	---	---
7	Floating	---	---

Table 2.1: Description of the Water Level Effect Attribute and the resulting depth ranges

interpolation. The Points raster (Figure 2.4c), which is composed of rocks, wrecks, buoys, beacons, etc., and the Polygon Raster (Figure 2.4d), which is composed of land, pontoons, shoreline construction, obstructions, etc., both have features whose depths may be outside the vertical bounds of the depth area and were created to enforce the requirements that the depths of these features, which may be either qualitative or quantitative, are represented in the final grid. The section below describes the process of giving a quantitative depth measurement for objects with only a qualitative measurement.

All ENC features that have the quantitative depth attribute, "value of sounding" (VALSOU), also have an attribute that describes the feature's depth qualitatively, the "water level effect" (WATLEV). WATLEV describes the depth of the object relative to the tidal cycle. NOAA's Hydrographic Surveys Specifications and Deliverables [22] gives a quantitative range for the WATLEV indices for both the Atlantic and the Pacific Oceans, as there are different hydrographic conditions on each coast, which are shown in Table 2.1.

Algorithm 1 Water Level to Depth

```
if ASV in Atlantic then
    Ocean_Scalar = 1
else
    Ocean_Scalar = 2
end if
if (Type == Land) then
    depth = −(MHW + 2)
else if WL == 2 then
    depth = −(MHW + 0.3048 ∗ Ocean_Scalar)
else if WL == 3 then
    depth = 0.3048 ∗ Ocean_Scalar
else if WL == 4 then
    depth = −(MHW + 0.3048 ∗ Ocean_Scalar)
else if WL == 5 then
    depth = −0.3048 ∗ Ocean_Scalar
end if
```

Figure 2.5: Algorithm to convert qualitative depth measurements to quantitative where positive depth values equate to deeper water.

These ranges were used in Algorithm 1 (shown in Figure 2.5) to provide conservative depth values for the WATLEV indices relative to MLLW and MHW for the features without quantitative depth information. In this work, the MHW value was chosen from the closest NOAA tidal station and MLLW and MHW were assumed constant for all areas in the ENC. (This may be inaccurate, especially in large scale charts or in areas where tidal range varies spatially. A more sophisticated method for determining tidal datums spatially is left for future work.) Additionally, all objects that are considered land (pontoons, shoreline construction, piles, floating docks, and land areas) were assumed to have a depth of 2 m above MHW.

After all depths were quantified and the rasters were created, the value of each cell in the interpolated grid was post-processed through the algorithm shown in Figure 2.6. Using this process, data from the original interpolated grid outside the ENC's bounds is removed (no data is denoted by a depth of -10 m) via the first check, the seafloor dependent objects in the interpolated grid are confirmed to be within the depth area's bounds (second step), and that the depths for all

Algorithm 2 Depth Resolution Process

Required Condition: All Rasters are the same size as the desired Map

```
for Index in Map do
  if Map[Index] is Inside ENC Bounds) then
    if DA_Shoal[Index] > Interp[Index] then
      Map[Index] = DA_Shoal[Index]
    else if DA_Deep[Index] < Interp[Index] then
      Map[Index] = DA_Deep[Index]
    else
      Map[Index] = Interp[Index]
    end if

    if −10 < Point[Index] < Map[Index] then
      Map[Index] = Point[Index]
    end if

    if −10 < Poly[Index] < Map[Index] then
      Map[Index] = Poly[Index]
    end if

  else
    Map[Index] = −10
  end if
end for
```

Figure 2.6: Algorithm to resolve the conflicting depth measurements using the rasters shown in Figure 2.4

seafloor independent obstacles are ensured to be in the final grid (third and fourth steps). The utility of the conflict resolution process is illustrated in Figure 2.7 where the depths are bounded by the depth area rasters and the artifacts crossing the UNH Pier from interpolation have been removed.

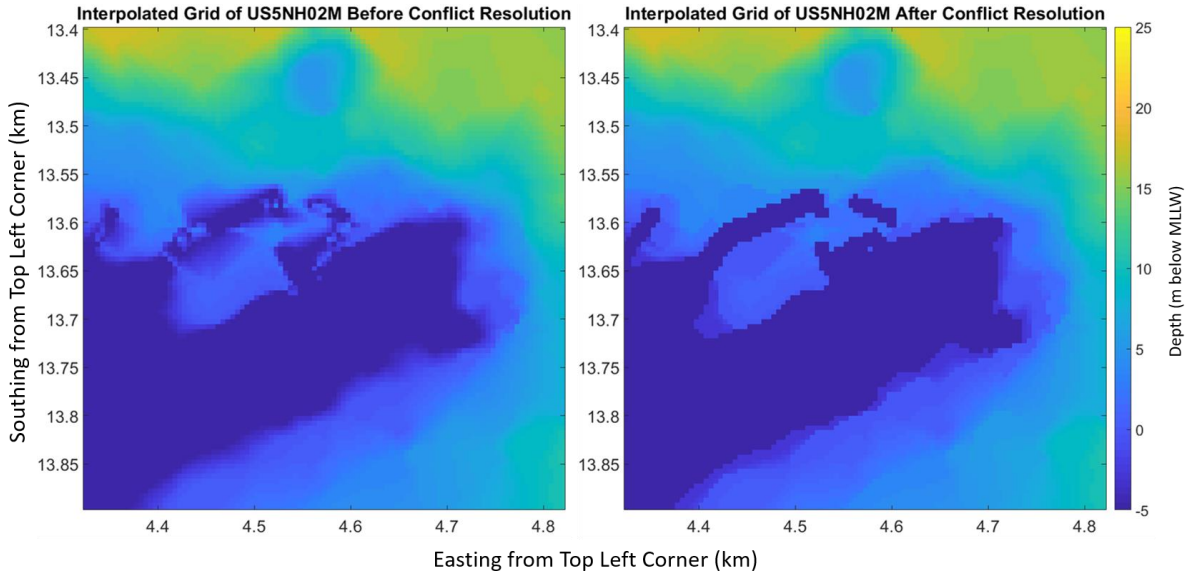


Figure 2.7: Artifacts from interpolation in the left image are corrected in the right image.

Section 2.1.3: Depth-Based A*

In this research, the cost function for determining the ASV's path was augmented from the classic A* implementation of minimizing the path length by also maximizing depth under the ASV using the previously described, ENC-derived grid. Adding a depth component to the cost function more accurately emulates how human mariners navigate, creating paths that stay to the central channel in shallow areas and travel directly to the desired location in deeper water. We call utilizing this novel cost function with A*, Depth-Based A*. Once the ASV is navigating in water deeper than 15 m, the depth-based cost is omitted in the cost function as deep water poses no threat to the ASV. The cost function, *G(n)*, for Depth-Based A* is shown in Equations 2.3 and 2.4:

$$G(n) = G(n\text{-}1) + \left((DP) + w * \text{Cost}_{\text{depth}}\right) \tag{2.3}$$

$$\text{Cost}_{\text{depth}} = \begin{cases} (DP)(15\text{-}AD) & AD < 15 \\ 0 & AD \geq 15 \end{cases} \tag{2.4}$$

where *G(n-1)* is the cumulative cost to get to the previous grid cell, *DP* is the distance from the cell *n* to its parent node (*n-1*) and the *AD* is the average depth between the nodes calculated by integrating the depths between the two nodes and dividing by the distance between them. The depth-based cost is weighted by *w*, which provides a tuning parameter to accommodate different size vessels as well as other factors including the seafloors characteristics (i.e., rocky or sandy bottom) or the additional sensors on the vessel (i.e. towed side scan sonar).

Figure 2.8 shows two missions planned with the Depth-Based A* algorithm while varying weighting of the depth-based cost, *w*. The two missions show five different weights: 0 (black), 0.1 (purple), 0.15 (white), 0.25 (red), and 0.5 (green). As the weight of the depth-based cost increases, the planned path becomes more conservative. A weighting of 0.15 for a vessel like the ASV Global C-Worker 4 [23] (4.0 m in length, 1.58 m in width) as originally constructed (i.e., not towing any

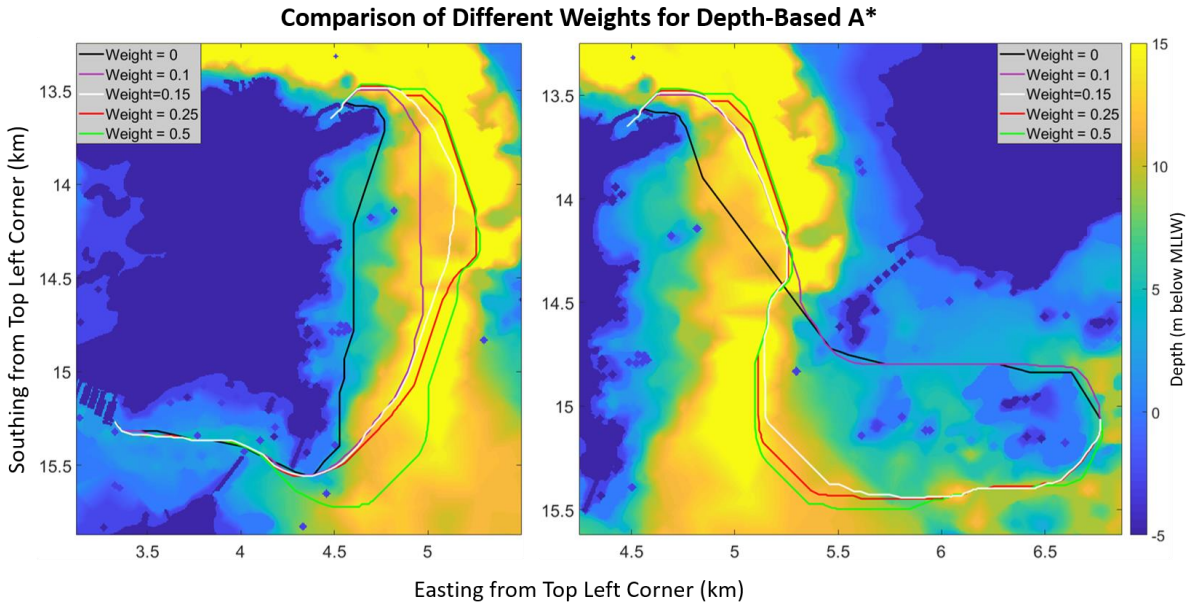


Figure 2.8: Comparison of different weighting for the depth-based cost in Depth-based A* for two different missions starting from the UNH pier.

additional equipment) was chosen as it allows the ASV to take a direct path while maintaining a safe depth under the boat.

Section 2.1.4: Mission Planner

Figure 2.9 illustrates the difference between the classic A* and Depth-Based A* mission planner, with an example mission planned using both methods. The classic A* algorithm utilizes a binary grid to indicate permissible or impermissible cells, whereas the Depth-Based A* algorithm utilizes the previously discussed ENC-based graph. These mission plans show how the addition of maximizing depth to the cost function gives a safer planned path than only minimizing path length, as it gives a wider safety margin to the border of the impermissible region.

The Depth-Based A* algorithm utilizing a cost function on an ENC-derived depth graph creates an environmentally aware mission plan that simulates the navigational strategies of human mariners. However, the marine environment is not static and a pre-generated mission plan does

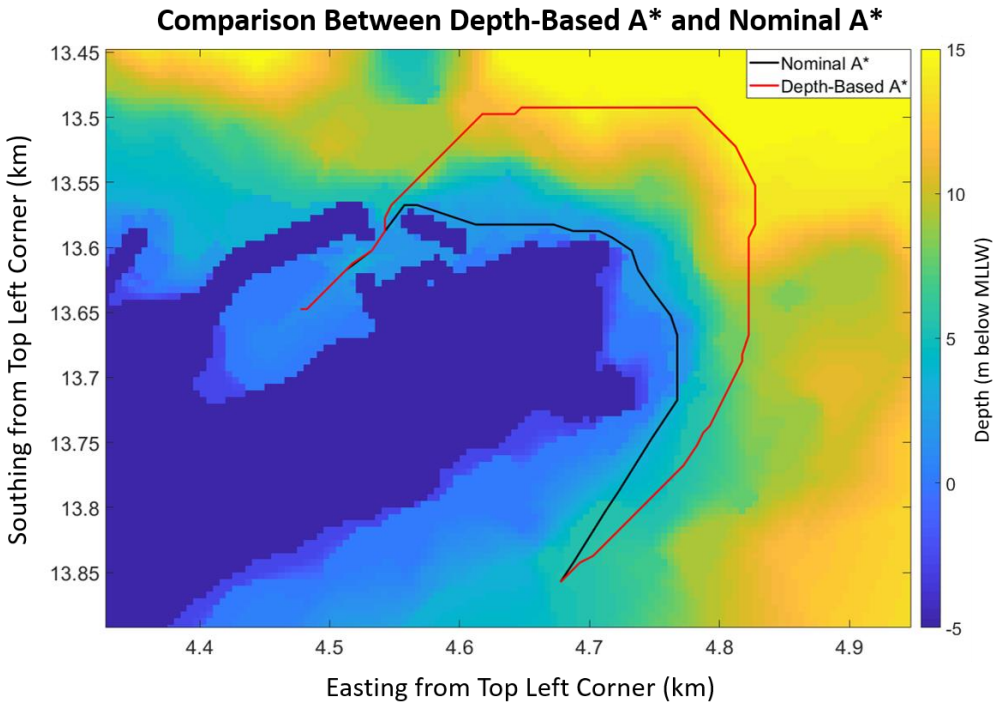


Figure 2.6: Comparison of A* mission plans utilizing a depth-based cost function (red) and minimum path length cost function (black).

not have the capacity to account for uncharted obstacles and other vessels. Therefore, in addition to the mission plan, a reactive obstacle avoidance component is necessary to guarantee the ASV's safety. The following section describes the chart-based reactive obstacle avoidance system developed in this work.

Section 2.2: ENC Derived Obstacle Avoidance

All reactive obstacle avoidance behaviors and applications have been developed in the Mission Oriented Operating System (MOOS) and the Interval Programming (IvP) Helm, an open source autonomy middleware and ship driving package [18],[19],[20]. The navigation and control of an ASV requires a current and desired position, along with the location of obstacles in the ASV's path. When the desired heading and speed have been established, a mechanism to convert these into rudder and thrust commands for the vehicle's propulsion and control system is required. These

components are implemented in the MOOS-IvP architecture and are described in detail in the following sections.

Section 2.2.1: MOOS-IvP Architecture

MOOS is a software package for robotics that consists of a set of libraries and executables running in a "publish and subscribe" architecture. This architecture works as a framework for communication between applications using a centralized database, referred to as the MOOSDB. MOOS's framework allows applications to publish and subscribe to data streams without knowledge of other applications. In the case of autonomous vessels, data streams can consist of the position and internal states of the vessel as well as information regarding the vessel's environment. Some pre-existing MOOS applications include a data logger (pLogger), a graphical user interface (GUI) for overseeing missions (pMarineViewer), a 3D vehicle simulator (uSimMarine), a controller to convert desired speed and heading to thrust and rudder angle (pMarinePID), and the IvP Helm [18],[19].

The IvP Helm is a MOOS process for ship driving in which the appropriate course of action results from the combination of separate behavior modules. Individual behavior modules each produce an "IvP Function", usually represented as a surface in polar coordinates whose azimuthal angle represents heading, radial distance represents speed, height indicates the utility of any combination of speed and heading, and whose peak indicates the optimal speed and heading for the vessel. These surfaces approximate each behavior's underlining utility function and are combined in the Helm's IvP Solver to determine the optimal vessel course of action. Some preexisting behavior modules include waypoint following, station keeping, and loitering [20].

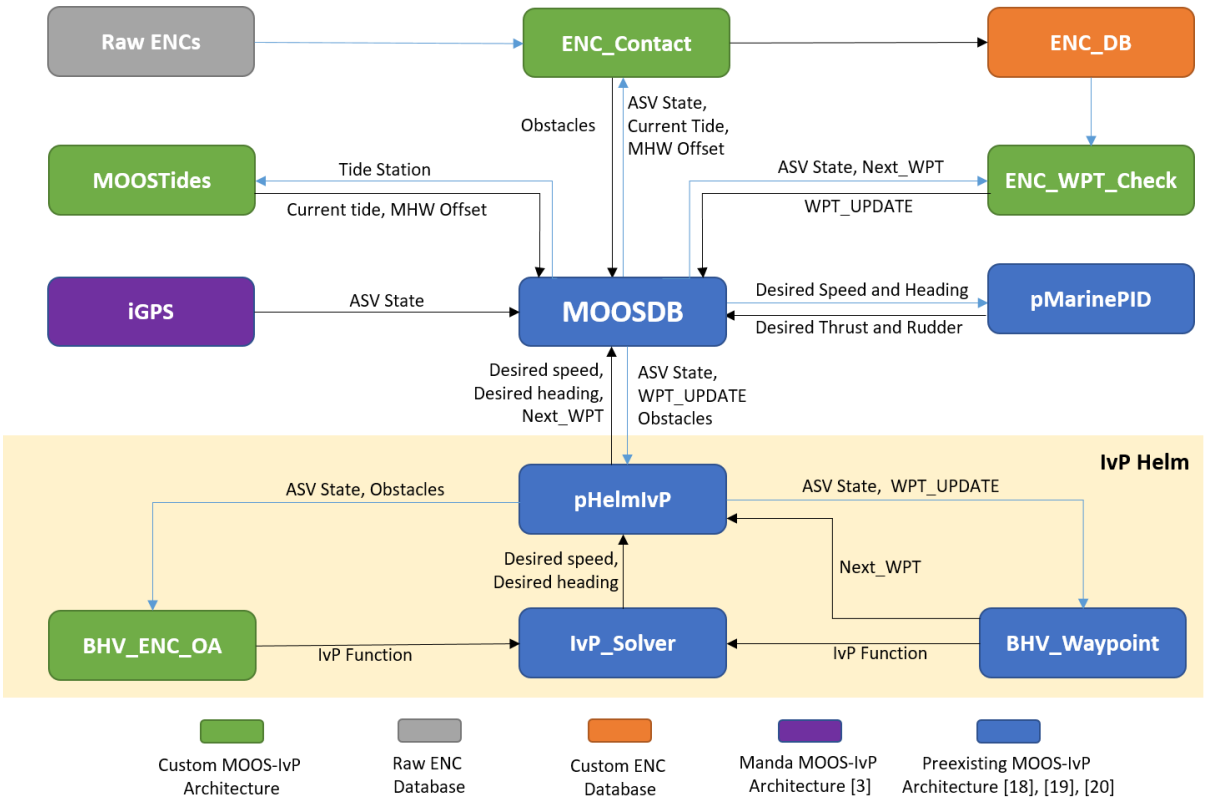


Figure 2.7: Flow chart diagraming the chart-based reactive obstacle avoidance system

In this research, custom MOOS applications and behaviors were developed for reactive obstacle avoidance. A topology of the utilized MOOS-IvP applications is shown in Figure 2.10, where the processes developed in this work are shown in green. In this architecture, as a pre-processing step, ENC_Contact converts an ENC covering the desired operating area into a spatial database that can be quickly searched in real-time, called ENC_DB. Inside ENC_DB, each feature is extracted and assigned a new attribute called “threat level”, which describes what threat the obstacle poses to the ASV. However, a full assessment of the threat requires a knowledge of the current tide, which can range globally from less than a meter to more than 10 m. A tidal prediction is provided using harmonic constituents from a user-provided tide station by the custom application MOOSTides. ENC_Contact then monitors the vessel’s position in real-time, identifies

obstacles in the near-field that pose a threat to the ASV while considering the tidal prediction, and publishes them to the MOOSDB. These processes are further discussed below.

The threats published by ENC_Contact are subscribed to by the obstacle avoidance behavior, BHV_ENC_OA. The vessel's instantaneous choice of speed and heading is established by combining the IvP Function from the waypoint behavior along with the one from BHV_ENC_OA as described in Section 2.2.4.

In addition to avoiding obstacles, the current waypoints, which are published by the waypoint behavior, are monitored in real time by ENC_WPT_Check. When a waypoint is considered unsafe or unreachable (e.g., it lies within an extended obstacle), the waypoint is skipped as the vessel approaches the obstacle's boundary. In this way, an operator may plan a mission with little regard to obstacles in the ASV's path and the vessel will make its best effort to complete the mission safely.

Section 2.2.2: MOOSTides

In the United States, nautical charts are referenced to MLLW allowing ships to safely navigate largely without accounting for tides. However, MLLW can be overly conservative for small vessels in areas with a high tidal range and a tidal model was used to more accurately assess the threat an obstacle poses to the ASV. MOOSTides publishes tidal predictions to the MOOSDB using the harmonic tidal model from Sam Cox's python library pytides [24] from a user specified tide station.

The real-time tidal prediction from MOOSTides is assumed to be equivalent for the entire region even though the tide may have a different phase and amplitude due to coastal effects. Although the harmonic tidal model implemented in pytides captures the effects of the gravitational

forces as well as the bathymetry of the local seafloor [25], it omits meteorological effects and tidal currents. Inclusion of more sophisticated tidal models is left for future work; however the inclusion of a more advanced tidal model might not be worth the cost of creating it unless the magnitude of the tidal range in the area of operations is large or significantly varies spatially.

Section 2.2.3: Threat Level

To address the vessel's dimensions and lack of recorded depth for some charted hazards in this thesis, a new attribute called "threat level" was developed. Threat level is defined using both vessel dimensions and depth information form the ENC, where it uses quantitative depth information (VALSOU) when available or qualitative information (WATLLEV) otherwise. The threat level attribute has a range from zero to five and is calculated using the flow chart given in Figure 2.11 and the algorithms in Figures 2.5, 2.12, and 2.13.

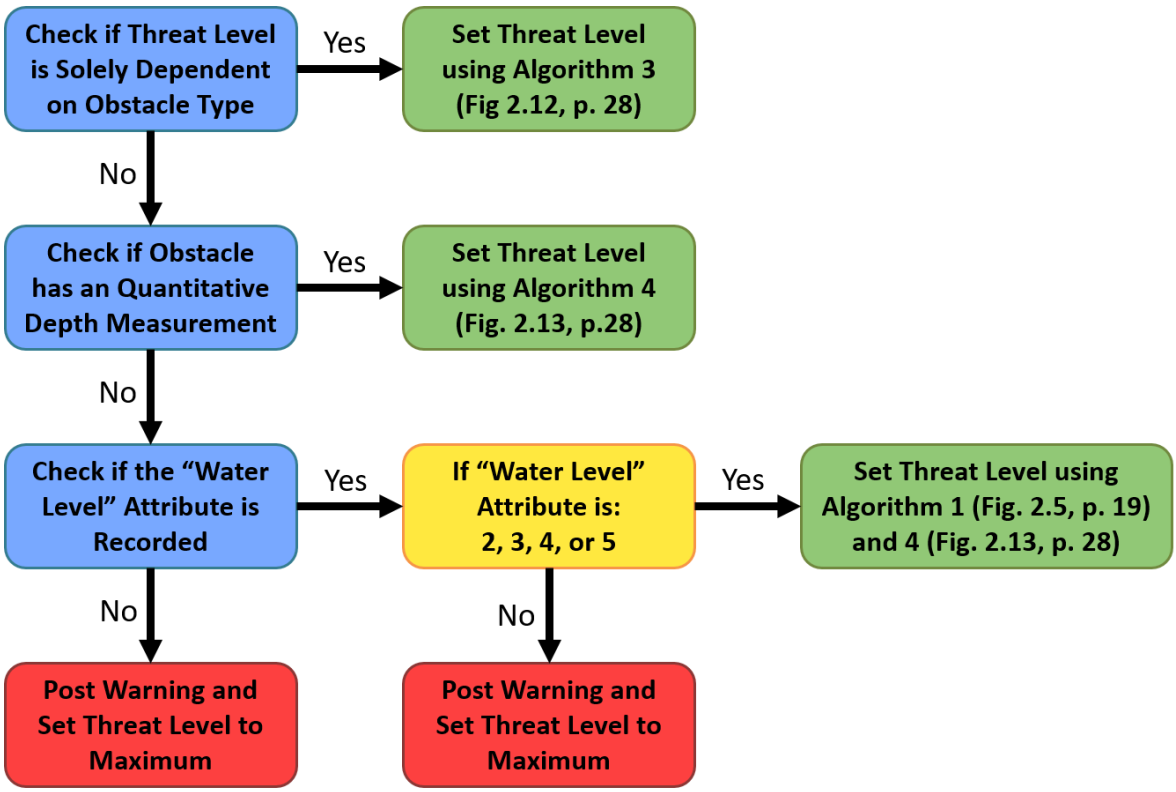


Figure 2.8: Flow chart describing how to determine an obstacle's threat level

The first step in determining the obstacle's threat level is to see if the object type is one without a quantitative depth (e.g., land, weeds/kelp, docks, buoys, beacons, etc.). If so, the threat level is determined by the algorithm in Figure 2.12. If the object can have an associated depth, the process then checks if the depth is explicitly given by the ENC. If so, the threat level is calculated by the algorithm in Figure 2.13. Otherwise, the method checks the qualitative depth information given by the feature's WATLEV attribute, which is described in Table 2.1. If the qualitative depth information is given, the threat level is set using the algorithm in Figure 2.5. If the first three checks fail or the WATLEV ID is 1, 6, or 7, (e.g., ones whose qualitative depths are not explicitly defined) then a warning is posted and the threat level of the object is set to four indicating that without more information, the object must be assumed to pose a great threat to the ASV.

Algorithm 3 Threat Level via Obstacle Type

if $(Type == Land)||(Type == Dyke)||(Type == Pontoon)$ **then**
  $Threat\_Level = 5$
else if $(Type == ShorelineConstruction)||(Type == Dock)$ **then**
  $Threat\_Level = 5$
else if $(Type == Buoy)||(Type == Weed/Kelp)||(Type == Beacon)$ **then**
  $Threat\_Level = 3$
else if $(Type == TopMark)||(Type == DayMark)$ **then**
  $Threat\_Level = 3$
end if

Figure 2.10: Algorithm to set threat level through obstacle type

Algorithm 4 Convert Depth (in meters) to Threat Level

Depth = MLLW_Depth+Current_Tide
if $(Depth <= (2 * Draft))$ **then**
  $Threat\_Level = 4$
else if $(Depth <= (3 * Draft))$ **then**
  $Threat\_Level = 3$
else if $(Depth <= (4 * Draft))$ **then**
  $Threat\_Level = 2$
else if $(Depth <= (5 * Draft))$ **then**
  $Threat\_Level = 1$
else
  $Threat\_Level = 0$
end if

Figure 2.9: Algorithm to set threat level using depth measurements

Section 2.2.4: ENC_Contact

To implement the real-time, reactive obstacle avoidance system, the data from ENCs are preemptively filtered to remove unnecessary information and translated into a spatial database for fast query by ENC_Contact. The native format of ENCs is S-57 [9], which organizes the spatial data by object type into different layers. The new spatial database, ENC_DB, created by ENC_Contact rearranges the ENC's layers from obstacle type into a single shape file composed of polygons. This organization is significantly more efficient for obstacle avoidance than the native S-57 format because, during each iteration, ENC_Contact only needs to open, filter, and react to a single layer instead of over 20.

The first step in creating ENC_DB, is to assign a threat level to each obstacle as described in Section 2.2.3. After the threat level for each obstacle is determined, each obstacle was buffered. As discussed in Section 2.1.2, due to the uncertainty of the charted objects with point geometry, points are buffered by one millimeter at chart scale giving them a diameter of two millimeter. Additionally, in order to assume the ASV is a point object, all objects are additionally buffered by equation 2.5 where TL is the obstacle's threat level, L is the ASV's length and W is the ASV's width. Depending on threat level, the buffer can range from the absolute minimum to assume the ASV as a point object (threat level zero) to three times the absolute minimum (threat level five).

$$B = \frac{(1+0.4*TL)}{2} * \sqrt{L^2 + W^2} \tag{2.5}$$

ENC_Contact also allows the user to set a minimum MLLW depth allowable for the ASV. This is accomplished utilizing an interpolated grid of the ENC using the method described in Section 2.1.3 for the Depth-Based A* mission planner. However, unlike the grid for the mission planner, the features used to create this surface only include soundings, piles, shoreline

construction, depth areas, land, pontoons, and docks. Once the grid is created, areas shoaler than the user's specified depth are polygonised and are placed into the spatial database (ENC_DB).

An example of the information extracted from the Portsmouth Harbor ENC (US5NH02M) is shown using MOOS-IvP's GUI, pMarineViewer, in the top image of Figure 2.14. (For reference, the bottom image of Figure 2.14 shows the nominal ENC display, typically shown through an

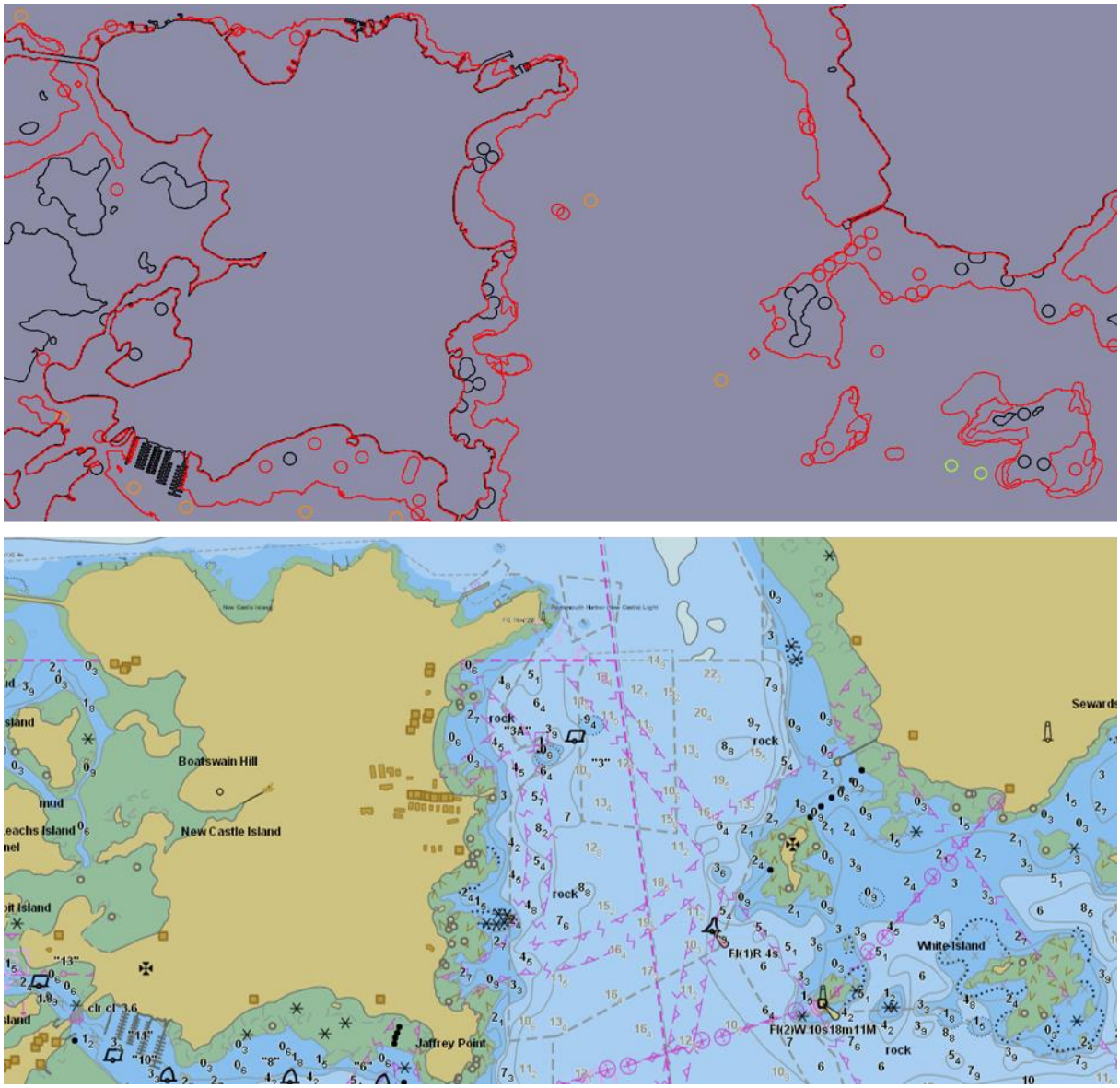


Figure 2.11: Information from the ENC_DB for the area around UNH's Pier in Portsmouth, NH shown using MOOS's pMarineViewer (top) and Electronic Chart Display and Information System (ECDIS) display for the corresponding ENC (bottom)

Threat Level	Obstacle Attributes	
	General Description	***Color***
0	No Threat	---
1	Small Threat	Yellow-Green
2	Medium Threat	Gold
3	Large Threat	Orange
4	Great Threat	Red
5	Land	Black

Table 2.2: Description and color on pMarineViewer of the threat level attribute

Electronic Chart Display and Information System or ECDIS display, for the corresponding area.) In the top image, the coastline is depicted with black lines, and the other features are color coded based on their threat level as described in Table 2.2. In this image, the tide set at zero meters and the draft of the vessel is 0.3 m. Obstacles with a threat level of zero (no threat) are not shown to avoid cluttering pMarineViewer.

Once ENC_Contact has translated the ENC into a useable spatial database, it then determines which objects are in the ASV's immediate vicinity, as determined from a predefined search area. This search area is defined as a square with each side having a length of $20L$ centered on the ASV's current position.

To determine safe headings, an angular-sweep algorithm is used to calculate the range of "utility" values in a full, 360-degree swath. Utility is defined between zero and 100, where a utility of 100 correlates to no current danger and zero is high danger. In this algorithm, rays of length $10L\sqrt{2}$ are projected every eight degrees from the ASV and each polygon in the search area is iteratively cycled through to check for intersections. If the ray intersects the polygon, the distance between the ASV and the polygon is determined and the utility of the resulting eight-degree sector

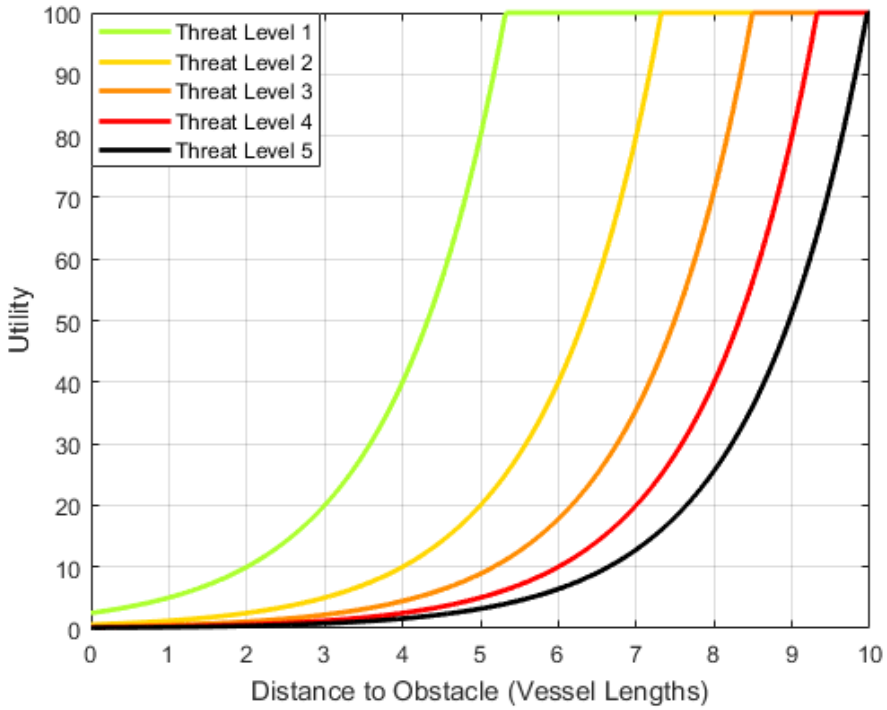


Figure 2.12: Distance vs Utility curves for threat levels one through five

is calculated and stored using Equation 2.6, where D is the distance to the obstacle in vessel lengths and TL is the obstacle's threat level. If the ray does not intersect the polygon, the utility for the resulting angle is 100. This process is repeated for each obstacle in the search area. If there is a previously stored utility for the angle from another polygon, then the lesser (more conservative) utility value is stored.

$$Utility = \min\left(2.5 * \frac{2^D}{TL^2}, 100\right) \tag{2.6}$$

Utility was defined in this way as the relationship between utility and distance should be concave up, the ASV should start penalizing headings farther away from the obstacle as threat levels increases, and an obstacle of threat level five should immediately penalize headings as soon as the ASV reaches 10 vessel lengths from the obstacle. The relationship between threat level, distance to an obstacle, and utility is shown graphically in Figure 2.15, where the distance to the object (defined in boat lengths) is shown on the x-axis and utility on the y-axis. Consider an

obstacle that is five vessels lengths from the ASV. Depending on the threat level of the obstacle, the utility can range from 80 (threat level one) to 8.9 (threat level three) to 3.2 (threat level five).

Using the angular sweep method as currently described, a ray that passes just outside of the obstacle would be classified as no threat and would be stored as a utility of 100. Therefore, to "soften" the edges of the polygon, the utility vector determined by the angular sweep algorithm was low-pass filtered and the minimum of the result before and after the low-pass filter (LPF) was taken as the new utility. This LPF was implemented using a moving average with a window size of five. In effect, the angle that each polygon subtends was expanded to further increase the ASV's safety margin as it effectively gives an angular buffer to the obstacle.

An example of the angular-sweep algorithm is shown in Figure 2.16. In the left panel of this figure, the ASV is assumed to be in the middle of the image with the rays emanating this location. Each intersection of the polygons is shown with blue dots. On the polar plot in the right panel, the resulting utility function is plotted with utility on the radial-axis and angle on the azimuthal-axis.

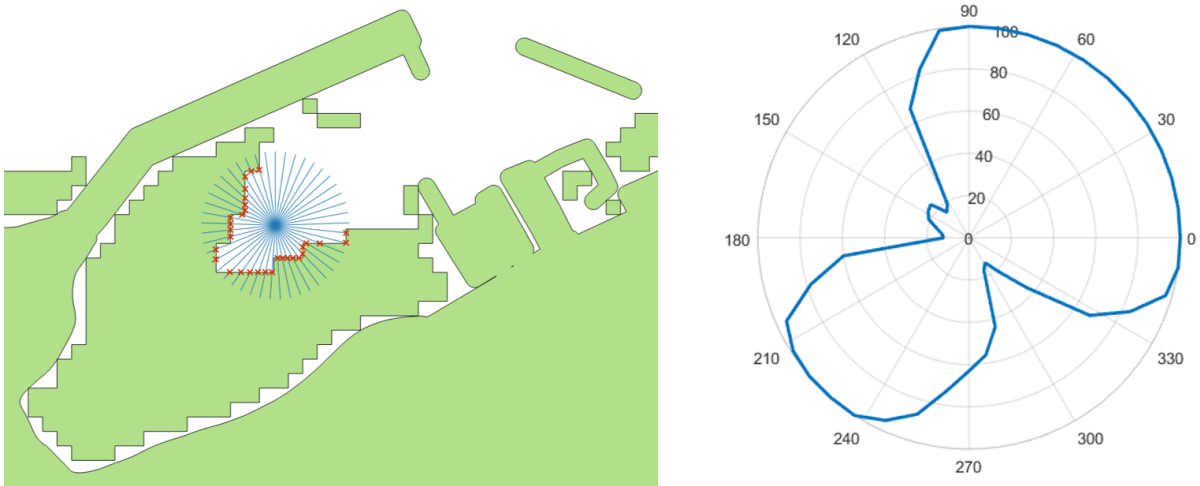


Figure 2.13: Example of the angular-sweep algorithm

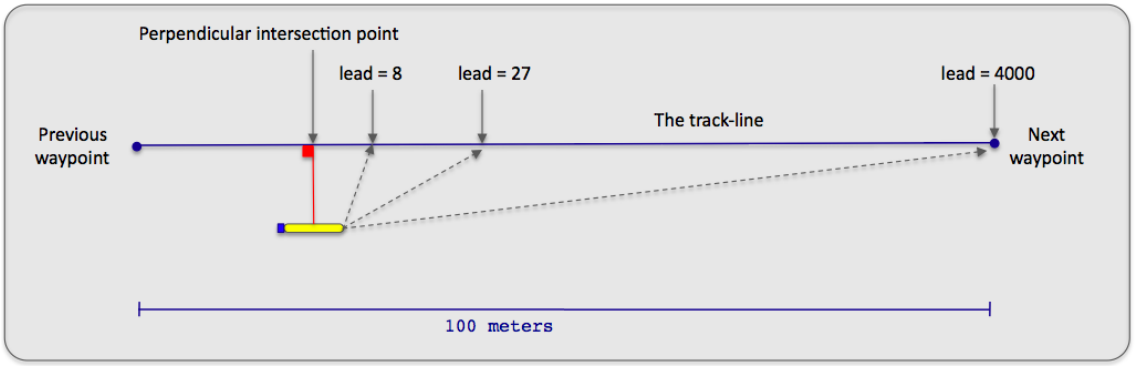


Figure 2.14: Example showing how the track-following works in MOOS's waypoint behavior, where the steering-point is determined by the lead parameter (i.e. the distance from the perpendicular intersection point toward the next waypoint). Used with permission from Mike Benjamin [18].

Finally, ENC_Contact also adjusts how strictly the waypoint behavior, MOOS's generic line-following program, follows the planned track when obstacles are inside the search area by adjusting the waypoint behavior's "lead" parameter. The lead parameter sets the location that the ASV dynamically steers toward, which is calculated using the distance (given by the value of the lead parameter) from the perpendicular intersection of the ASV's current location and the trackline as shown in Figure 2.17 [18].[4] Utilization of the lead parameter results in the ASV aggressively approaching the trackline when the perpendicular distance is large and gradually when it is close.

The result of adjusting the lead parameter is that the ASV only attempts track-following when it is safe to do so. This is defined as when there are no obstacles in the ASVs proximity or the ASV is driving away from the obstacle and is father than the safety distance (SD), defined in Equation 2.7, away from the obstacle. In this research, the value of the lead parameter is defined using Equations 2.7 and 2.8 where D is the distance to the closest obstacle on the current control iteration, ΔD is the change in distance to the closest obstacle since the last control iteration, MTL

[4]If the value of the lead parameter is greater than the distance to the next waypoint from the perpendicular intersection, then the location that the waypoint behavior's IvP Function steers is towards the next waypoint (as shown in Figure 2.17).

is the maximum threat level for all obstacles inside ENC_Contact's search area, and *SD* is a threshold around an obstacle defined as too close to attempt track following. The value of the lead parameter varies from a MOOS default of eight to a value of 100 near a threat level 5. In future research, Equations 2.7 and 2.8 might be adjusted to account for larger vessels, as the parameters may be insignificant in reference to the vessels size.

$$SD=3*L+0.5*MTL \tag{2.7}$$

$$lead=\begin{cases} 8 & \Delta D > 0 \; and \; D > SD \\ MTL * 20 & otherwise \end{cases} \tag{2.8}$$

After the angular sweep algorithm is completed for each obstacle in the ASV's search area, ENC_Contact publishes a comma separated list describing the utility for each candidate 8 degree heading choice. This message is published to the MOOSDB at 5 Hz and is subscribed to by the ENC-based reactive obstacle avoidance behavior.

Section 2.2.5 - Obstacle Avoidance

Once the nearby objects have been identified and unsafe heading choices are denoted, the next step is to determine the desired speed and heading for the ASV. In this research, the optimal speed and heading is calculated by optimizing the IvP Functions for the native waypoint behavior and a new behavior called ENC_OA. This new behavior subscribes to the heading utility report from ENC_Contact. With this new behavior, which penalizes unsafe headings, and the waypoint behavior, the IvP Solver determines the optimal speed and heading pair for the ASV.

In MOOS-IvP, each behavior produces an IvP Function, which is a polar surface that is defined over its domain (e.g, speed and heading for the waypoint behavior and heading for ENC_OA). A graphical representation of the waypoint behavior's IvP Function is shown in Figure

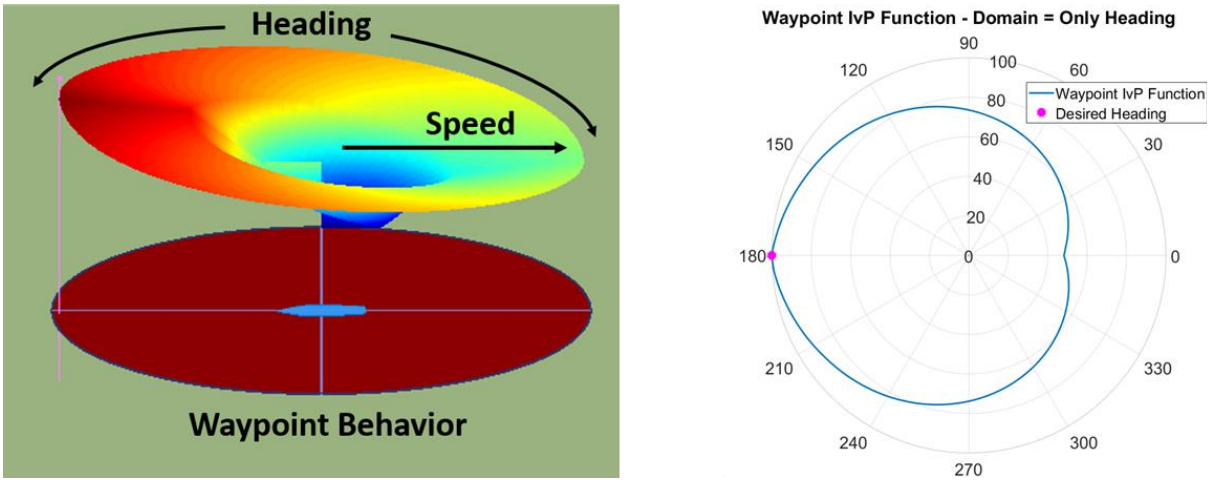


Figure 2.15: MOOS's waypoint behavior's IvP Function for a westward waypoint. In the left image the heading is shown in the azimuthal direction, speed in the radial direction, and utility with height and color. In the right image, a simplified IvP Function for the waypoint behavior is shown (excluding speed from the IvP Function's domain) where the heading is shown in the azimuthal direction and utility in the radial direction.

2.18. In the left image, the height indicates the utility of each combination of speed and heading and the peak indicates the optimal choice. The right image of Figure 2.18 shows the same IvP Function without the utility of speed (i.e., the domain only considering heading). This simplified representation of the waypoint's IvP Function is useful when illustrating the combination of the two IvP Functions as heading is the control variable in the ENC_OA behavior.

The IvP function of ENC_OA allows for any combination of heading and speed other than those that lead toward threats. ENC_OA uses the obstacle report from ENC_Contact, which gives the utility of the obstacles that intersected in the search area in eight-degree increments (after a LPF), and linearly interpolates between the angular increments to create its IvP Function.

Additionally, if the ASV is within three boat lengths from an obstalce, ENC_OA biases the utility function towards the ASV's current heading. Biasing the utility function increasingly penalizes making radical changes to the desired heading as drastically changing heading while the ASV is near obstacles can lead to dangerous situations. The new utility (U) from the heading bias

equation, given in Equation 2.9, linearly biases the previously calculated utility (PrevU) towards the current heading (H), which allows the ASV to avoid obstacles in a safer manner by severely penalizing radically changing the desired heading away from the ASV current track.

$$U=PrevU*\left(1-\frac{|H-\angle|}{360}\right) \tag{2.9}$$

ENC_OA allows for the ASV to avoid obstacles along the desired trackline in real time. However, if the planned mission has waypoints that reside inside obstacles in the ENC, the ASV will not be able to complete its desired mission, even with ENC_OA, and will require human intervention. Therefore, another MOOS application was developed, ENC_WPT_Check, to address this issue.

Section 2.2.6: Waypoint Checker

Reactive obstacle avoidance behaviors provide ASV path adjustments between predetermined waypoints; however, due to faulty mission planning some waypoints may not be reachable with a navigable/realistically safe path. ENC_WPT_Check addresses this issue by determining if the desired waypoint is navigable and if it is not, skipping it once the ASV has reached a user defined distance from the obstacle. The utility of ENC_WPT_Check is shown in the scenario in Figure 2.19 where the red polygons are obstructions and all waypoints reside inside of these regions. The image shows the path that the simulated ASV traveled and using ENC_WPT_Check, the ASV successfully skipped all waypoints without running into the high threat obstruction. Although the planned path has waypoints that terminate inside high threat obstacles, ENC_WPT_Check allows a user to plan a mission near an irregular shoreline or

obstructions and be confident that the ASV will not run aground without requiring the planning of a complex survey pattern.

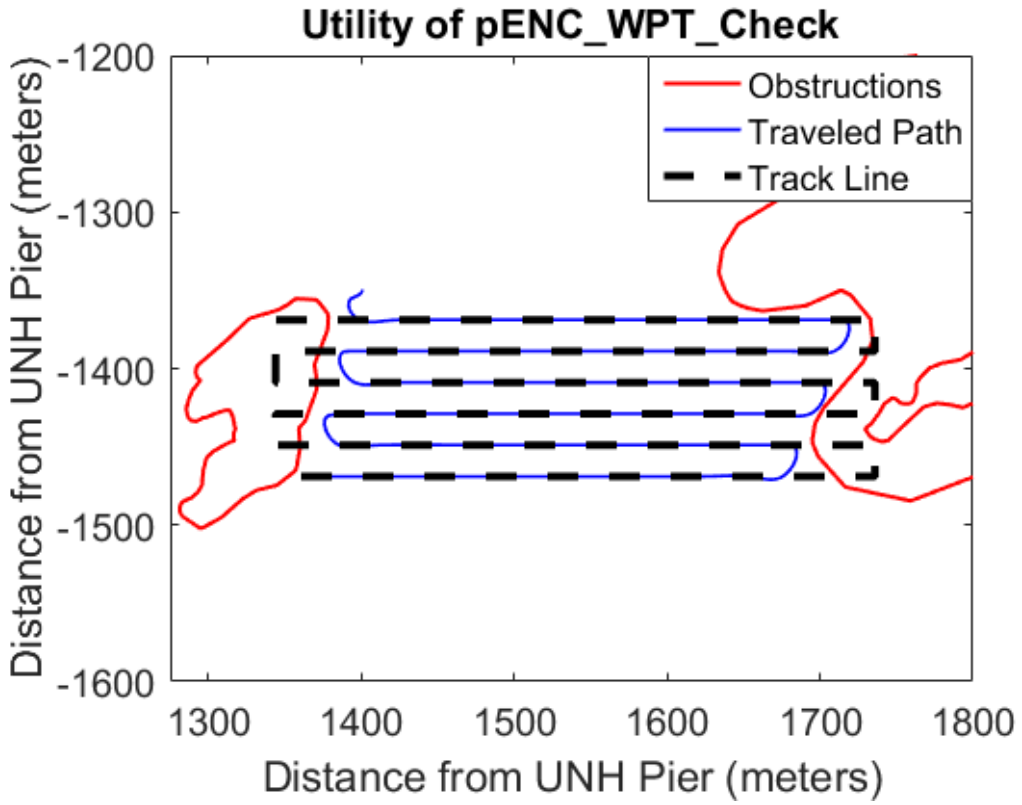


Figure 2.16: Scenario in Portsmouth, NH harbor showing the utility of ENC_WPT_Check, where the ASV skips the unsafe waypoints on its desired path. Displayed is the ASV's traveled path (blue), the obstructions (red) and the track line (black dotted line).

Chapter 3

Results

3.1: Depth-Based A* Mission Planner

To illustrate the features of the Depth-Based A* mission planner, missions were planned on multiple ENCs. The first mission was planned from the UNH pier facility up the Piscataqua River to a pier near Prescott Park. To create the mission plan, the Depth-Based A* algorithm was given the start and end points along with the interpolated grid, as described in Section 2.1.2, generated from the US5NH02M ENC. In Figure 3.1, an overview of the entire mission is shown

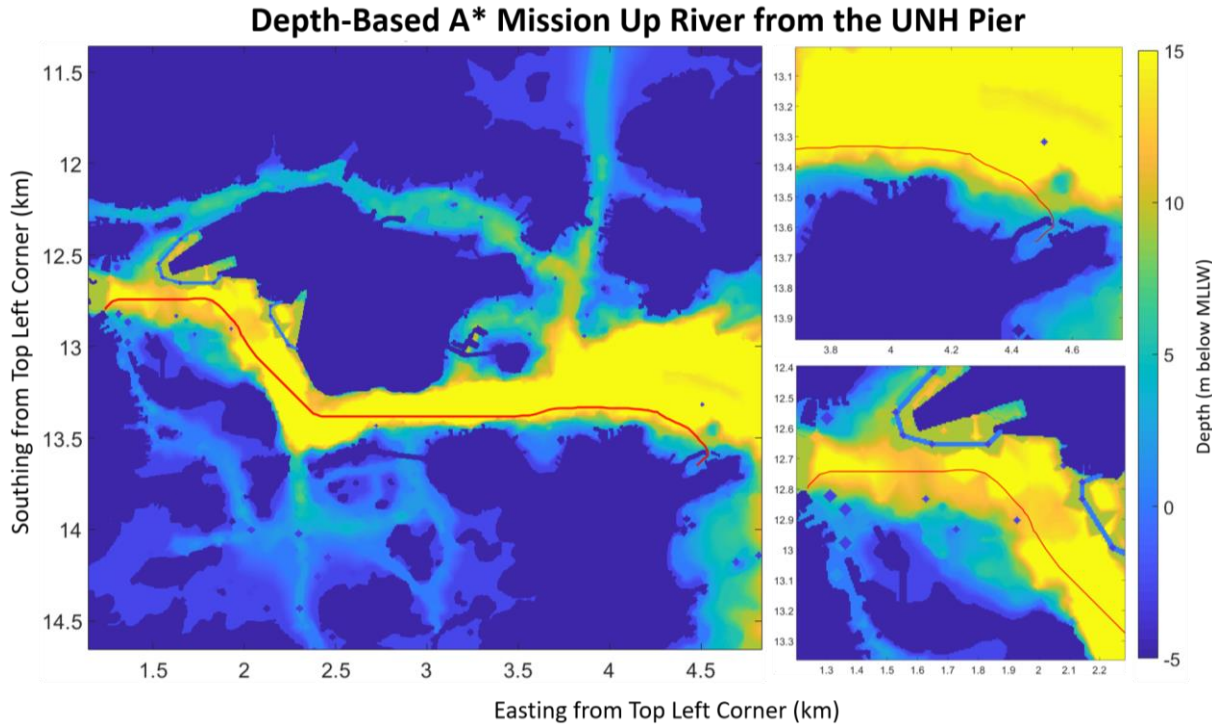


Figure 3.1: Depth-Based A* Mission plan (red) from the UNH Pier to a pier near Prescott Park overlaid on the interpolated grid. The left image is an overview of the mission and the right two images zoom into either the start (top right) and the finish (bottom right).

in the left panel, the top right panel depicts the start near the UNH pier, the bottom right panel portrays approximately the last third of the mission, and the color scale is defined between -5 m above MLLW (i.e., land) and 15 m below MLLW. In these images, the color scale does not depict water deeper than 15 m as any depth greater does not accumulate any depth-related cost.

The planned mission safely avoids the pier and the breakwater before driving toward the channel. Once the depth below the boat's keel is deeper than 15 m, there is no additional depth-based cost and the mission planner follows nominal A*, where the planned path is the shortest distance between the start and end points through the admissible space. Near the end of the planned path, the river becomes shallower and, in these sections, the mission planner balances the two components of the cost function by planning a path to safely navigate through the shoaler areas without unnecessary maneuvers.

In some situations the cost of traveling through shallow, non-dangerous waters is less costly than traveling through deeper water. To illustrate this, Figure 3.2 shows a different mission planned on the US5NH02M ENC that begins at the UNH pier and finishes near the Isle of the Shoals in Maine. The top right panel shows the planned path leaving the UNH pier and avoiding the breakwater before navigating towards the channel and the bottom right image shows the end of the mission near the Isle of Shoals.

However, unlike the previous mission, the planned path does not exclusively drive in the deep water (shown as yellow in the figure). In a few cases, such as the middle lower-third in the top-right image and the middle of the bottom-right image (highlighted in the figure with the arrows), it was less costly to travel through shallower, but non-dangerous water, than to drive

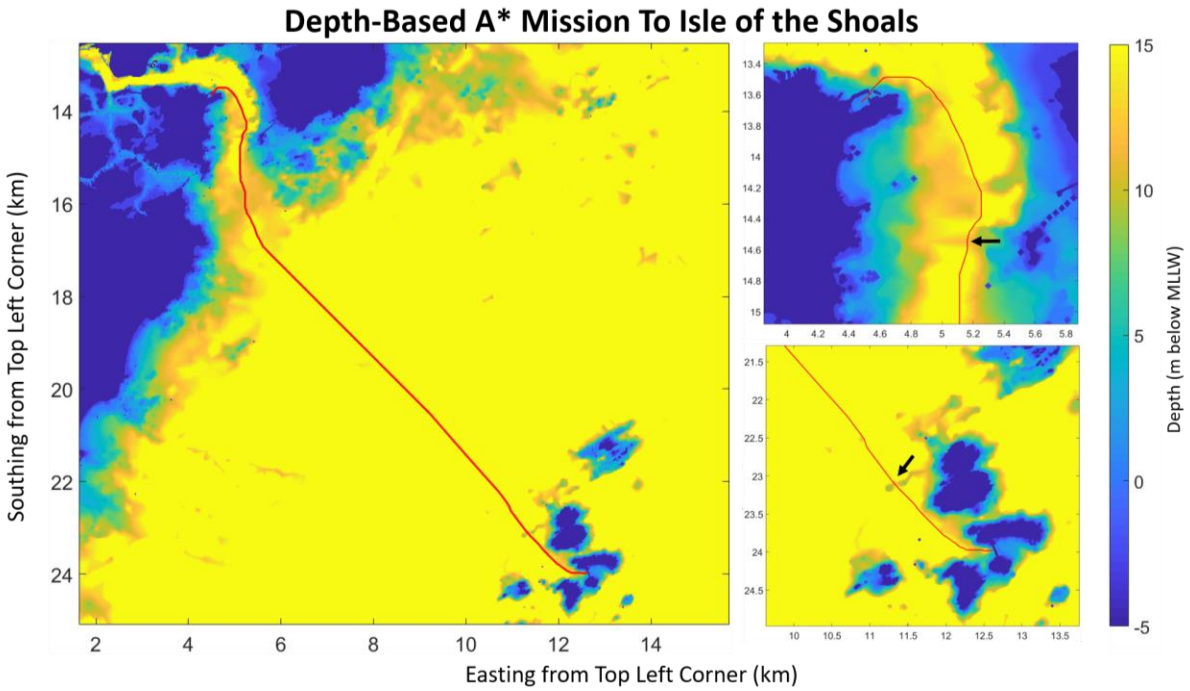


Figure 3.2: Depth-Based A* Mission plan (red) from the UNH Pier to the Isle of the Shoals in Maine overlaid on the interpolated grid. The left image is an overview of the mission and the right two images zoom into either the start (top right) and the finish (bottom right).

around it. This is a key aspect of the Depth-Based A* mission planner, where it balances taking the most direct route with risk of grounding.

Figure 3.3 illustrates another mission planned with the Depth-Based A* algorithm in Boston Harbor (US5MA11M). This mission starts at a pier near the Old North Church and ends at a pier at the Winthrop Yacht Club. After leaving the pier near the Old North Church, the path lies almost exclusively in the channel for the first half of the mission. However, approximately half way through the mission, the path leaves the channel and drives through shallower areas to effectively "cut the corner". The planned mission drives approximately 2 km less than the alternative route following the channel; however, it still remains in the safe, deeper sections of this shallower area.

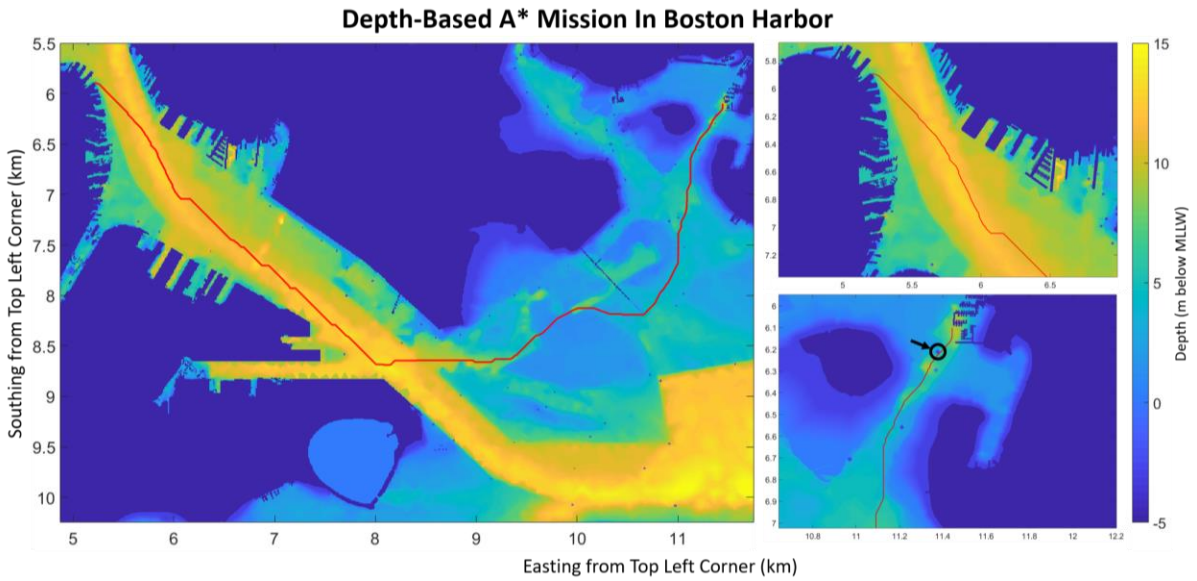


Figure 3.3: Depth-Based A* Mission plan (red) in the Boston, Massachusetts Harbor overlaid on the interpolated grid. The left image is an overview of the mission and the right two images zoom into either the start (top right) and the finish (bottom right).

As the planned path approaches the end of its mission (near the pier at the Winthrop Yacht Club), the path drives very close, approximately 2.5 m, to the lateral buoy (highlighted with an arrow in the bottom right panel). This is still safe as the obstacle is buffered; however, the close proximity to a mapped obstacle shows a limitation in the mission planner. Driving this close to the buoy in non-ideal conditions may be unsafe. This behavior is due to the masking of seafloor independent obstacles in the conflict resolution process, described in Section 2.1.3, as the obstacles, the buoy in this case, may be near relatively deep water and the water surrounding the buffered obstacle may not be heavily penalized. Therefore the path may be planned on the edge of the permissible regions. Using the reactive component of this research would mitigate these issues.

These three missions clearly show that the Depth-Based A* mission planner avoids known obstacles and shoal areas from the ENC while attempting to stay to the channel, much like a human mariner. However, as shown in the third mission, the planned path can still drive near obstacles in

the ENC. The reactive component of this research further increases the ASV's safety and should be used in conjunction with the Depth-Based A* mission planner.

3.2: Reactive Obstacle Avoidance

The reactive obstacle avoidance algorithms were tested in both the MOOS simulation environment, using a MOOS application called uSimMarine to simulate the ASV's states (position, speed and heading), and in the field using a Seafloor Systems EchoBoat [26] (shown in Figure 3.4).

3.2.1 Non-ENC (Synthetic) Obstacles

For the initial tests, a synthetic ellipse with a major axis of 8 m and a minor axis of 5 m was placed in the middle of the ASV's planned path. For these tests, the ASV traveled clockwise along the planned rectangular path, encountering the ellipsoidal obstacle with an increasingly higher threat level after each circumnavigation. Figure 3.5 depicts the path traveled by the simulated ASV while avoiding the synthetic obstacle. In these simulations, the dimensions of a Seafloor Systems EchoBoat (1.68 m, 0.79 m, 0.28 m) as well as an ASV Global C-Worker 4 (4.0

Figure 3.4: Profile view of the Seafloor Systems EchoBoat

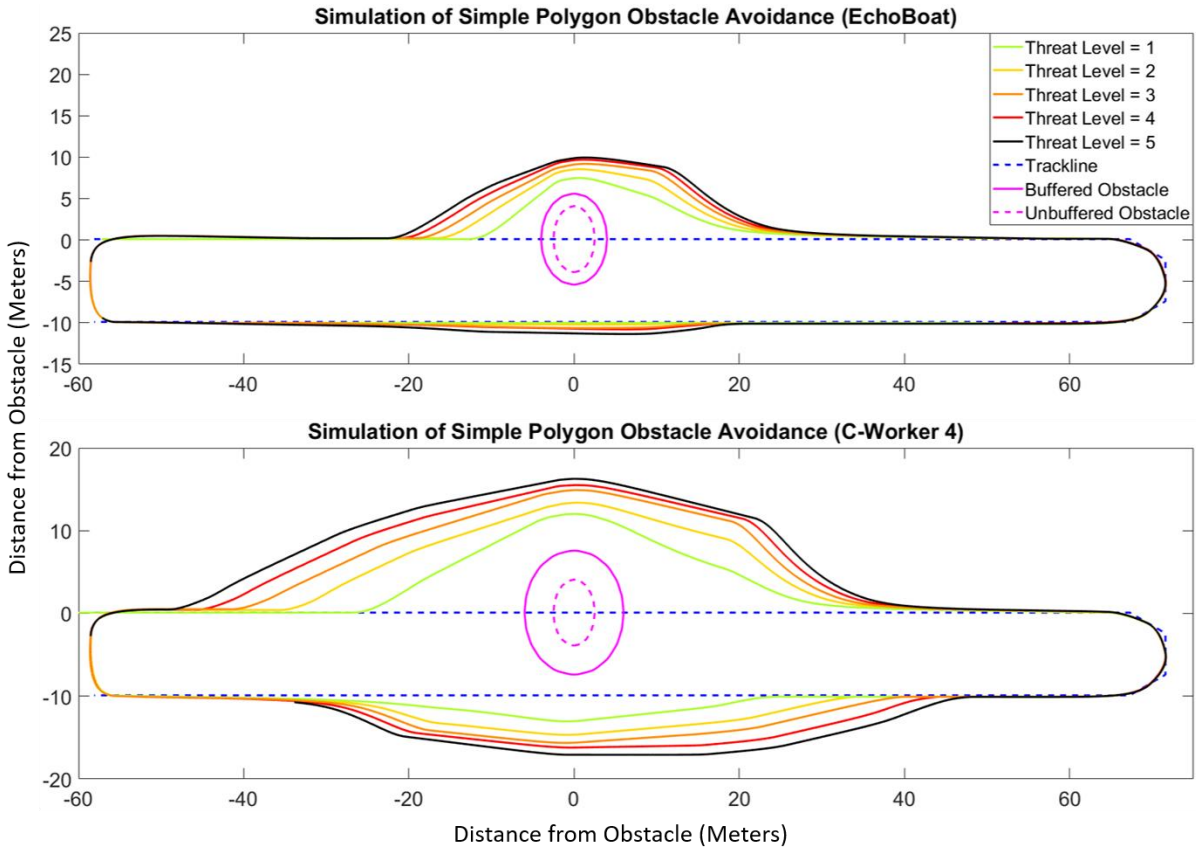


Figure 3.5: Mission around an ellipse (pink) with varying threat levels during simulation where the color of the trackline defines the obstacle's threat level. The top image's simulates a Seafloor Systems EchoBoat and the bottom image simulates an ASV Global C-Worker 4.

m, 1.58 m, 0.4 m) were used and the desired speed was set to the EchoBoat's nominal speed of 1.5 m/s. The pink, solid-line ellipse in these images is buffered version of the synthetic obstacle (pink, dotted line). This buffer, 1.3 m for the EchoBoat and 3.0 m for the C-Worker 4 (calculated using Equation 2.5 with a threat level 0 and the vessel's dimensions), allows for the comparison of the paths of different size vessels around the obstacle for all threat levels as the ASV is represented as a point and not as multi-dimensional boat of various sizes.

In each of these simulations, the ASV avoids the ellipse by at least 1.8 m or 4.4 m for the simulated EchoBoat and C-Worker 4 (as shown in Table 3.1). As the threat level of the obstacle

	Minimum Distance Between ASV and the Synthetic Ellipse Obstacle (meters)	
Threat Level	**Simulated EchoBoat**	**Simulated C-Worker 4**
1	1.8 m	4.4 m
2	2.9 m	5.8 m
3	3.4 m	7.3 m
4	4.0 m	7.9 m
5	4.2 m	8.7 m

Table 3.1: Minimum distance from the ASV to the ellipse-shaped obstacle while varying vessel length in simulation

increases, the simulated ASV gives the ellipse an increasingly wider berth as the penalty function (given in Equation 2.6 and shown graphically in Figure 2.15) is inversely proportional to the obstacle's threat level squared.

After the ASV passes the obstacle, it does not immediately prioritize track following. Instead, it waits until the distance between the obstacle and the ASV is greater than the safety distance (defined in Equation 2.7). Once the ASV has passed this threshold, ENC_Contact modifies the waypoint behavior's lead parameter, which adjusts the location that the ASV steers toward (as described in Section 2.2.3).

This same scenario was field tested with the Seafloor System EchoBoat. In the field, the test consisted an obstacle of threat levels 1, 3 and 5. The resulting tracklines are shown in Figure

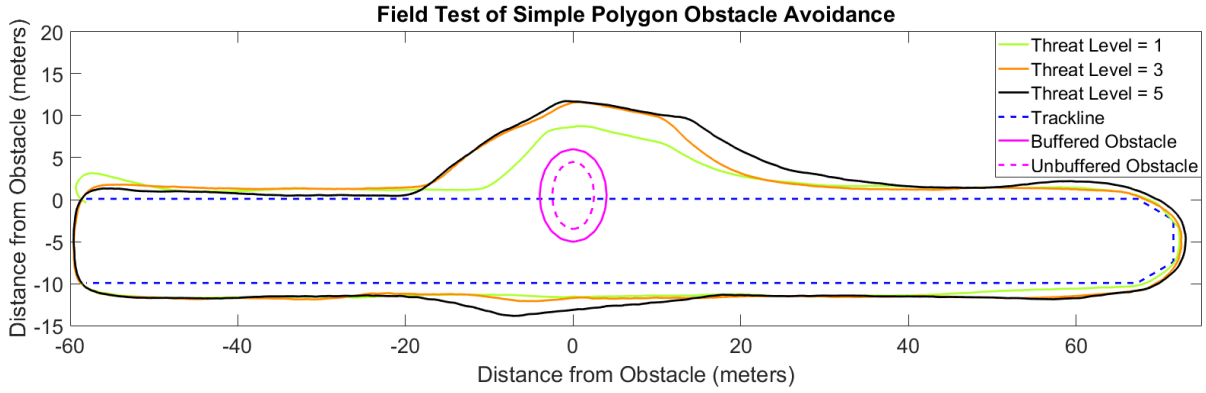


Figure 3.6: Mission around an ellipse (pink) with varying threat levels from field data, where the color of the trackline defines the obstacle's threat level.

Threat Level	Minimum Distance Between EchoBoat and the Synthetic Ellipse (meters)
1	2.6 m
3	4.8 m
5	5.3 m

Table 3.2: Field data on the minimum distance from the EchoBoat to the synthetic ellipse

3.6 where the color of the tracklines relates to the threat level of the synthetic ellipse. During the field tests, the EchoBoat traveled no closer than 2.6 m to the ellipse. The remaining minimum distances between the ASV and the ellipse are shown in Table 3.2.

There are a few noticeable differences between the simulated and field results. The main difference is the EchoBoat's path for threat level 3 and 5 are very similar, since the desired heading for the ASV with the threat level 5 obstacle oscillated between choosing a northerly or southerly path around the obstacle over a distance of 2.1 meters. Once the ASV hits a critical point, the desired heading remains to the north and the ASV follows a similar path to the threat level 3 trackline. Although these oscillations are not ideal, the ASV avoided all polygons safely. Further discussion of this behavior is in Section 4.1.2.

3.2.2 ENC-Based Obstacles

The reactive algorithm was also tested in situations where the ASV's planned path went through an obstacle denoted in the ENC, specifically the breakwater near the UNH pier and the UNH pier itself, requiring ENC_OA to provide a new, safe path. The results shown in the following tests are from data collected in the field.

The EchoBoat successfully avoided the breakwater using only the reactive method in two different directions: driving into the cove (Figures 3.7 and 3.8) and leaving the cove (Figures 3.9 and 3.10). The first mission started on the far-side of the breakwater with reference to the UNH

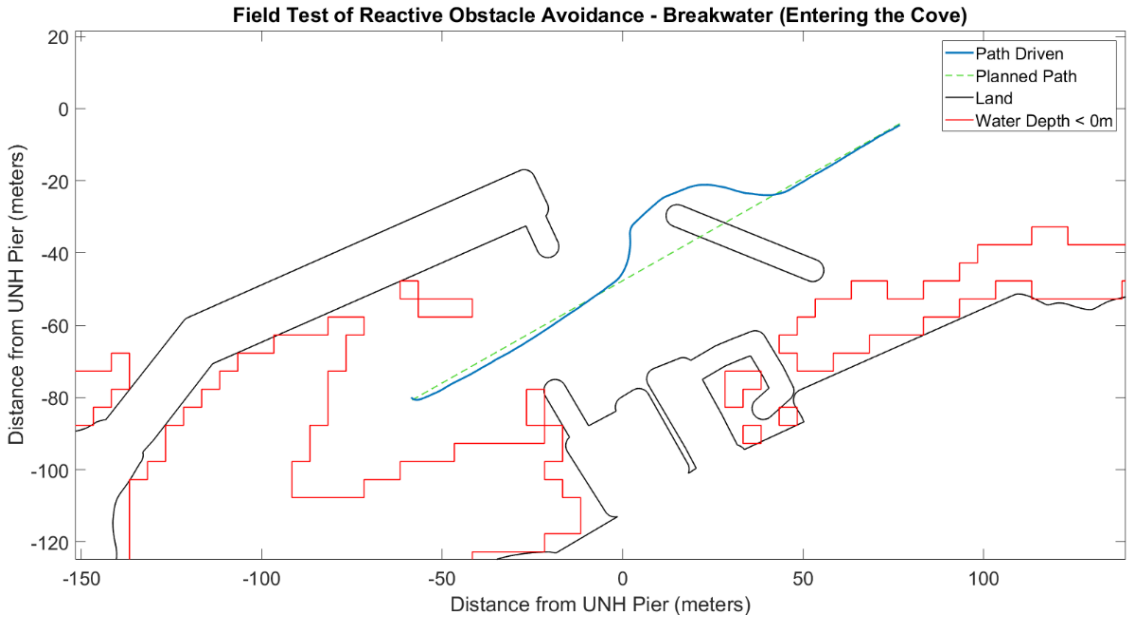


Figure 3.7: Plan view of a mission driving around the breakwater near the UNH Pier using ENC_OA to avoid the breakwater. In this mission, the ASV drove from the far-side of the breakwater (with reference to the UNH pier), around the breakwater, to inside the pier's cove on the path shown in blue.

pier, in which the EchoBoat's planned mission was to line-follow through the breakwater. (A plan view of the mission is shown in Figure 3.7.)

Due to the angular profile of the breakwater in reference to the planned path, the southern edge of the breakwater came into the search area first. The images in Figure 3.8 show snapshots of the IvP Functions (left), where the colored lines denote the Heading IvP Functions and the stars denote the desired (black) and current (red) heading, and a plan view of the mission, where the ASV's current position is denoted with a yellow dot (right). The top images of Figure 3.8 clearly show that headings that lead toward the breakwater (southwest) are penalized by ENC_OA (orange IvP Function in the left image) and the desired heading shifts approximately 30 degrees to the northwest from the planned path. As the EchoBoat navigates around the breakwater, the penalized headings shift from the southwest to the southeast and once the ASV navigates around the

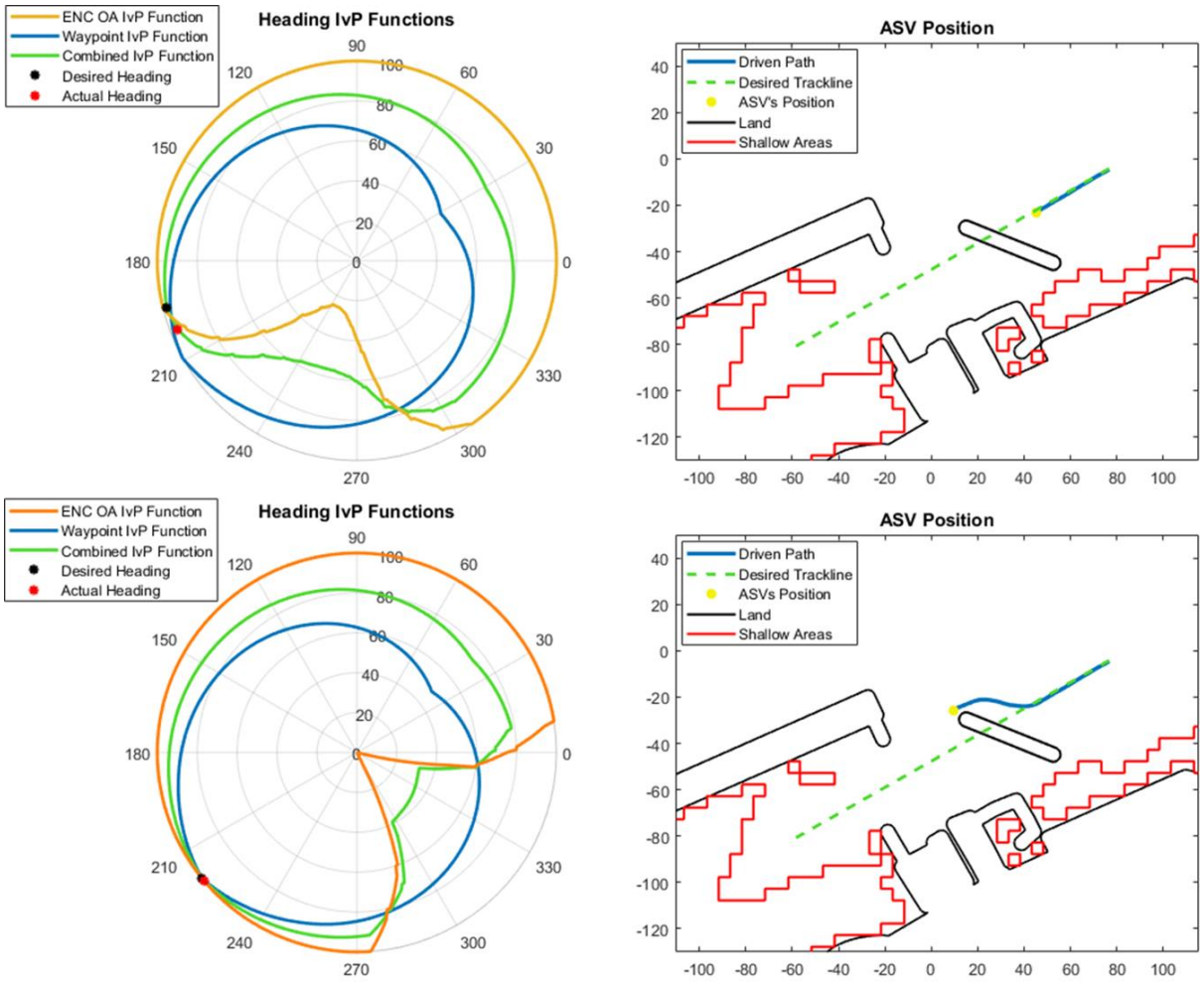


Figure 3.8: Snapshots at the two critical junctions of the mission where the EchoBoat drove around the breakwater while line-following where the left images show the heading based IvP functions for ENC_OA (orange line), the waypoint behavior (blue line) and the combined IvP Function (green line) as well as the current and desired heading (red and black stars respectively) and a plan view of the mission (right images) where the ASV's position is marked with a yellow dot at the time of the IvP Functions.

dangerous area, the EchoBoat begins prioritizing track-following (bottom panels of Figure 3.8). During this mission, the EchoBoat drove no closer than 3.0 m away from the buffered breakwater.

In the next mission, the ASV avoided the breakwater after starting on the near-side with reference to the UNH pier (a plan-view of the mission is shown in Figure 3.9). During this mission, the ASV avoided the breakwater towards the south instead of the north (unlike the previous

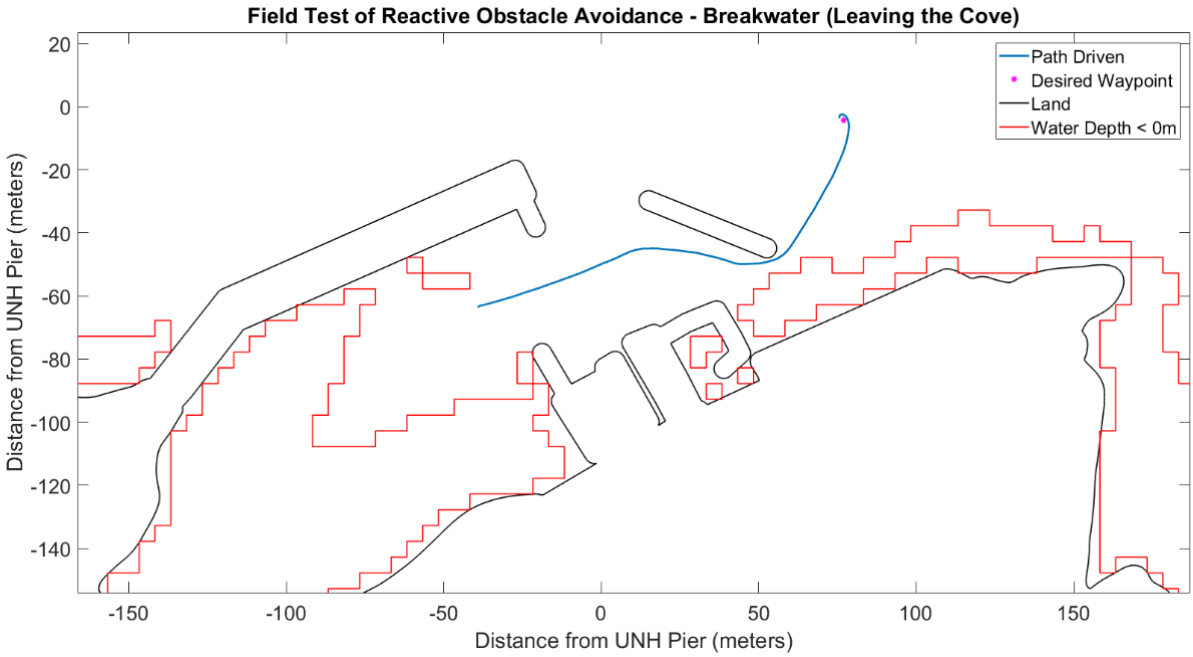


Figure 3.9: Plan view of a mission driving around the breakwater near the UNH Pier in the field using ENC_OA to avoid the breakwater

mission) as only the northwest corner of the breakwater was visible to ENC_Contact as the breakwater entered into the ASV's search area. The left panel of Figure 3.10 shows that the penalty of driving around the breakwater to the north, due to ENC_OA (orange line) changing the combined IvP function (green line), was enough to force the EchoBoat to attempt to circumnavigate the breakwater to the south towards Fort Constitution. Once the EchoBoat drove to the southeast corner of the breakwater, the ASV was forced to split the gap between the breakwater and the shallow water polygon (bottom panels of Figure 3.10). As a result, it drove very close to both the breakwater (1.1 m at its closest point) and the shallow water (3.6 m at the closest point). Although the ASV drove very close to the buffered breakwater and shallow water, it successfully avoided all obstacles in its environment and safely arrived at its desired waypoint without human intervention. In these two examples, the ASV safely completed its mission by using

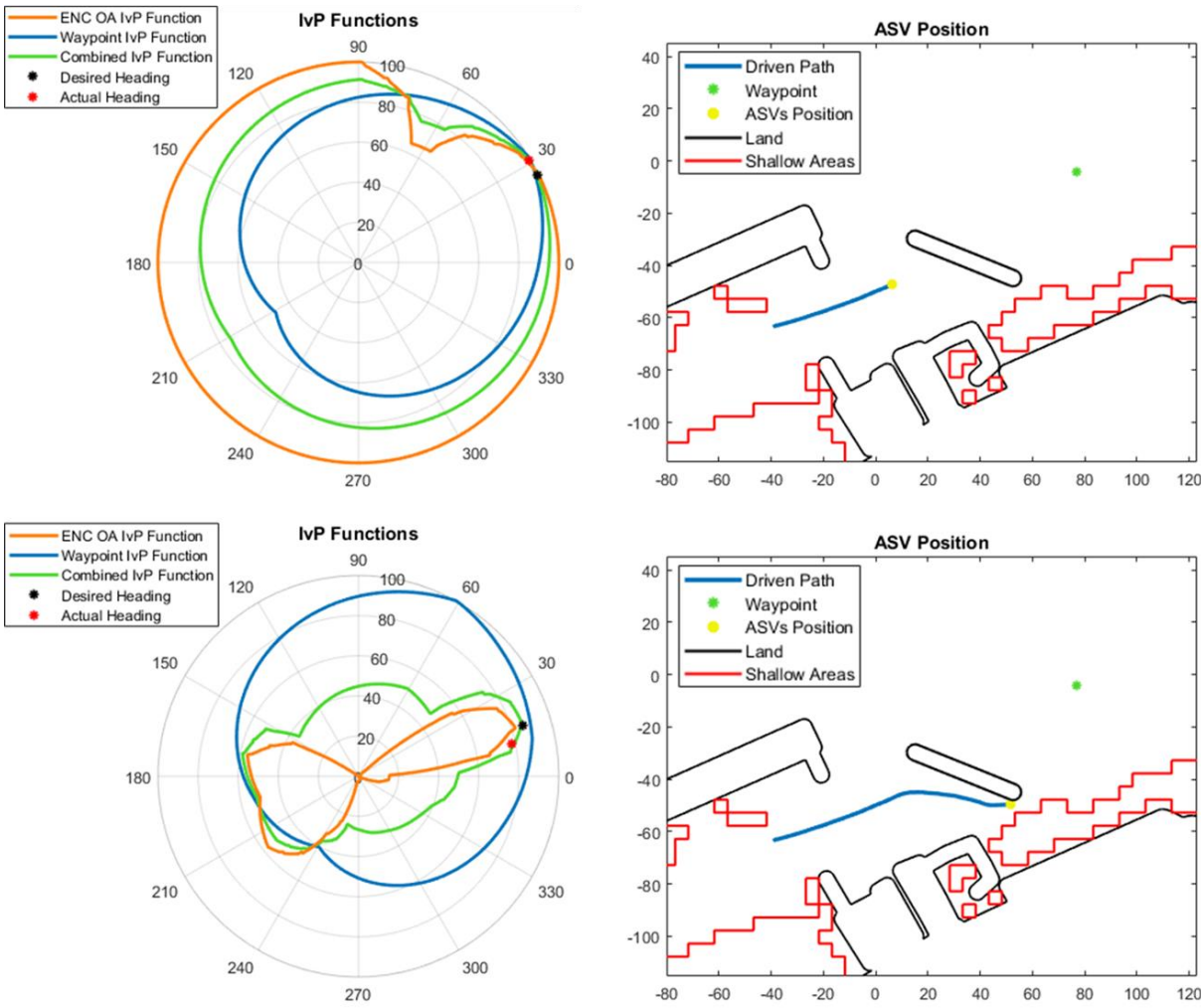


Figure 3.10: Snapshots at the two critical junctions of the mission where the EchoBoat drove around the breakwater where the left images show the heading based IvP functions for ENC_OA (orange line), the waypoint behavior (blue line) and the combined IvP Function (green line) as well as the current and desired heading (red and black stars respectively) and a plan view of the mission (right images) where the ASV's position is marked with a yellow dot at the time of the IvP Functions.

ENC_OA to adjust its path around unsafe mission plans in real-time using both waypoint navigation and line following.

In the next mission, the EchoBoat's planned path drove straight through the pier as shown in Figure 3.11. Similar to previous missions, the ASV stayed on the desired path while it was safe, but when it was hazardous, the ASV left the planned path and avoided the pier. A snapshot of the mission as the ASV approaches the pier is shown in Figure 3.12 where the IvP functions illustrate

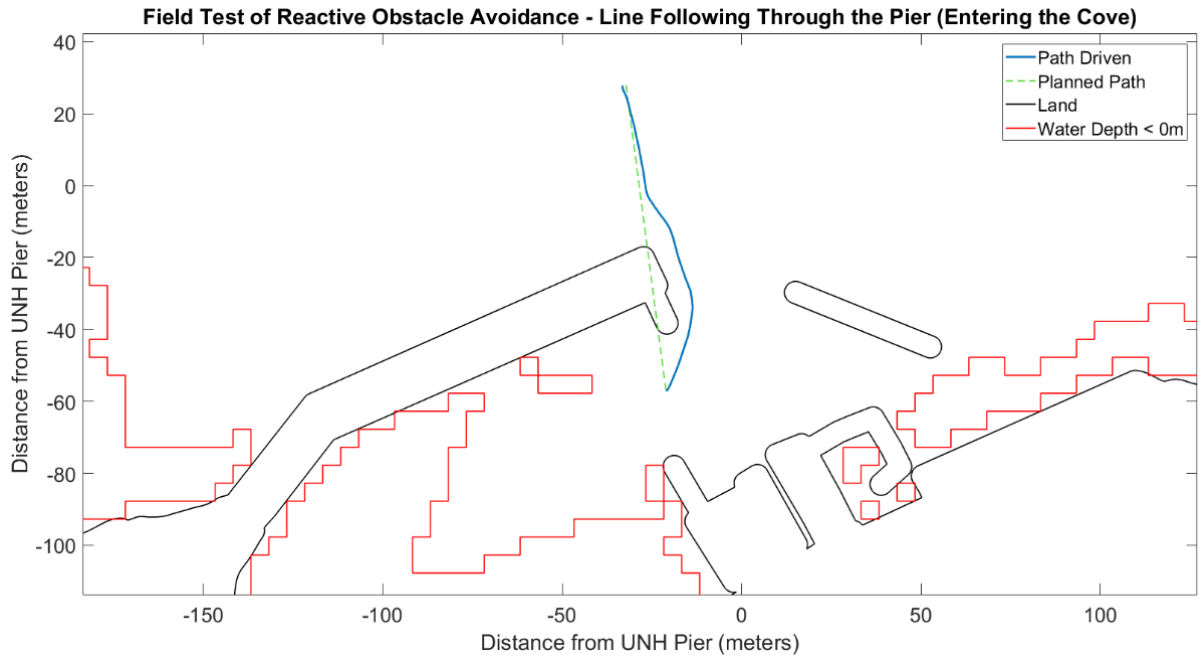


Figure 3.11: Plan view of a mission driving around the UNH Pier in the field using ENC_OA. The planned path is given with the green, dashed line and the path driven by the EchoBoat is shown in blue.

that the headings leading toward the UNH pier are penalized. As shown previously, once the EchoBoat safely navigated around the pier, the ASV returned to its desired trackline. Even though

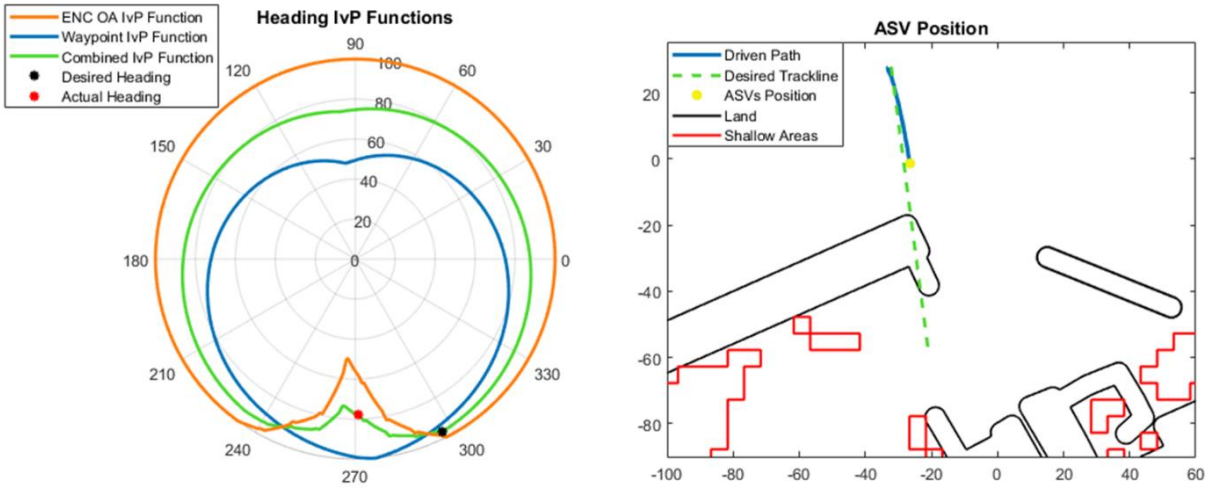


Figure 3.12: Snapshot during the mission line-following while avoiding the UNH pier while in the field when ENC_OA starts to push the desired heading of the EchoBoat to the southeast to avoid the pier. In the left image, all IvP functions are shown with ENC_OA in orange, the waypoint behavior in blue, and combined IvP function in green as well as the current and desired heading with a red and black star respectfully.

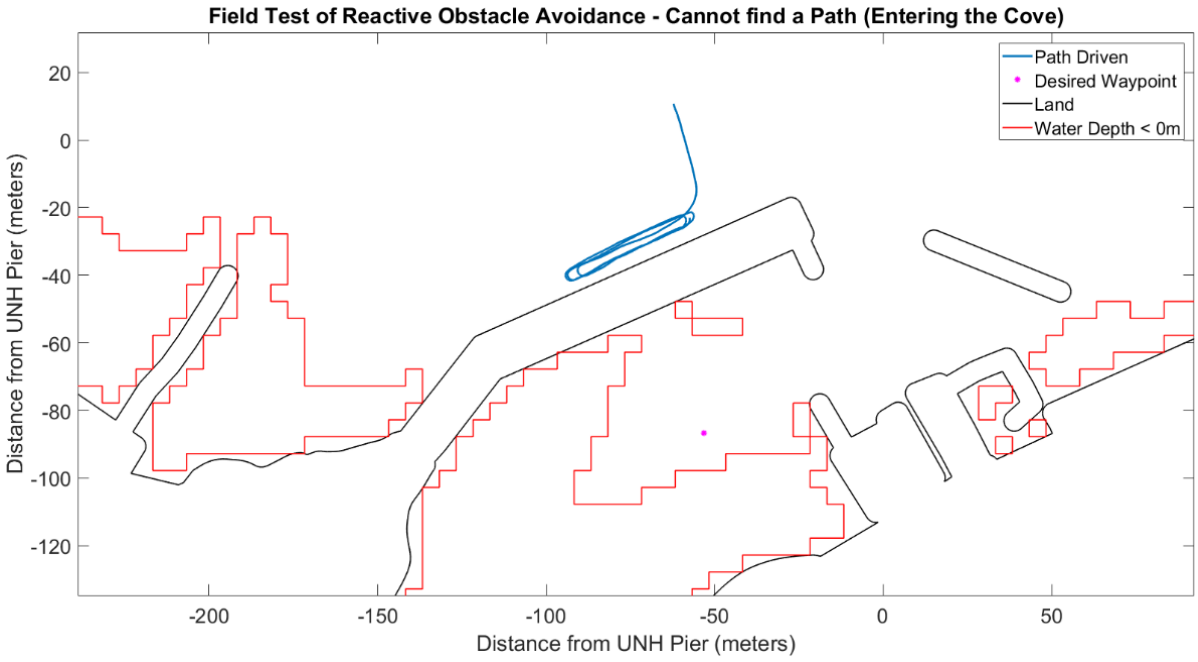


Figure 3.13: Plan view of a mission driving towards the UNH Pier in the field where the EchoBoat got stuck in a local minimum. The path driven by the EchoBoat is shown in blue and the desired waypoint is shown with a pink star.

the pier is a non-uniform obstacle, ENC_OA allowed for the ASV to safely reach its desired location, while driving no closer than 3.4 m away from the pier, despite poor mission planning.

In some situations, the reactive mission planner can get stuck in local minima. An example mission is shown in Figure 3.13, where the EchoBoat attempted to navigate from one side of the pier to the other. In this mission, the EchoBoat oscillates between heading choices to achieve the waypoint and those that are penalized by the presence of the pier. This cycle was repeated *ad infinitum* as the EchoBoat was stuck in a local minimum and the mission had to be canceled. Snapshots of the heading IvP Functions, shown in Figure 3.14, illustrate the critical junctions of the ASV path as it got stuck in a local minimum. Although this behavior is not ideal, the ASV came no closer than 8.3 m away from the pier. This result illustrates that reactive behaviors alone may be insufficient for reliable navigation and that additional behaviors that look with a greater

field of interest, such as running Depth-Based A* when stuck in a local minimum, is required. Implementation of these behaviors are left for future work.

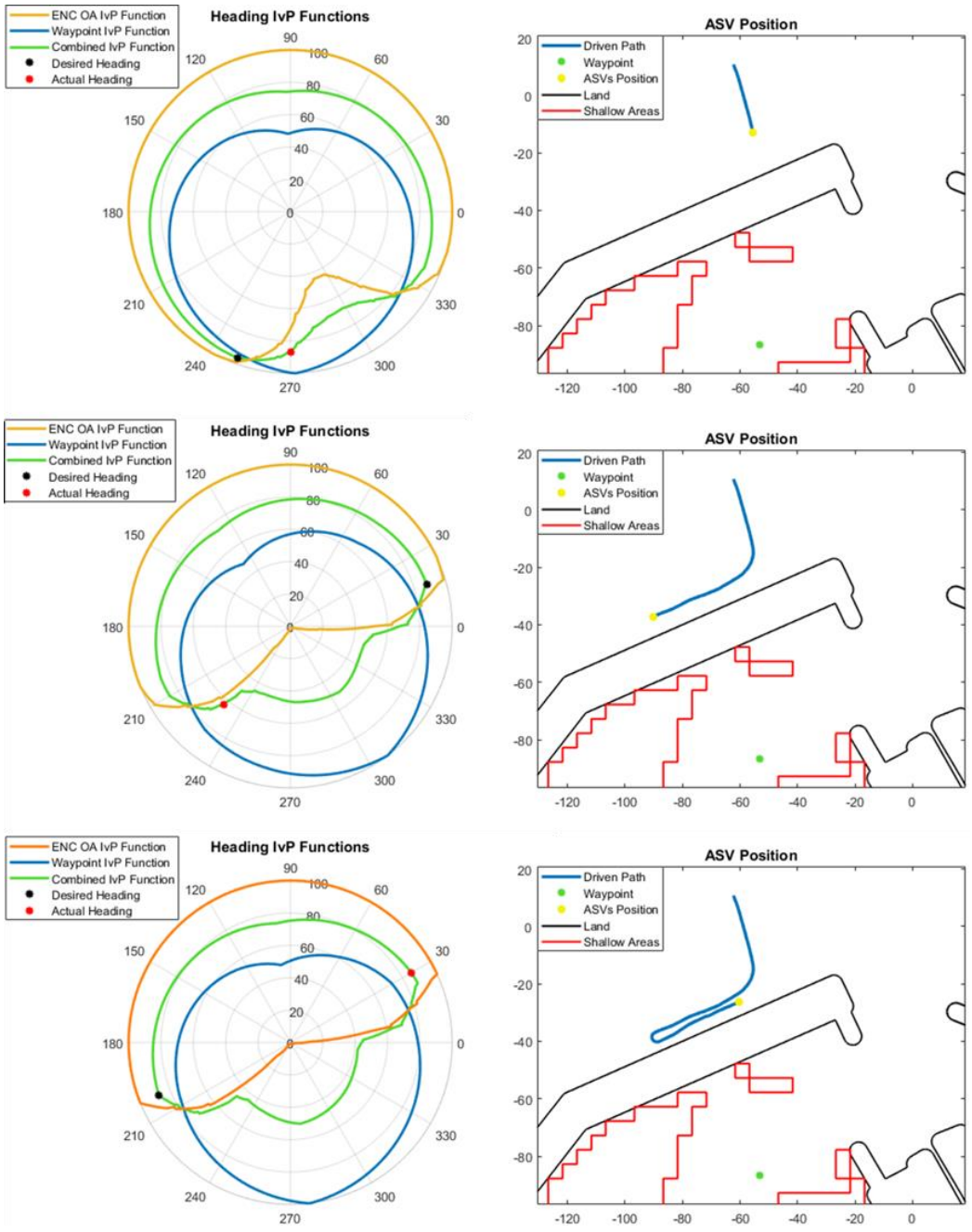


Figure 3.14: Snapshots at the three critical junctions of the mission where the EchoBoat got stuck in a local minimum. The left images show the heading-based IvP functions for ENC_OA (orange line), the waypoint behavior (blue line) and the combined IvP Function (green line) as well as the current and desired heading (red and black stars respectively) and a plan view of the mission (right images) where the ASV's position is marked with a yellow star at the time of the IvP Functions.

These manufactured scenarios of purposely driving at the pier, breakwater and synthetic ellipse show that the reactive obstacle avoidance procedures keep the ASV from hitting charted obstacles. Although charted obstacles can typically be avoided through mission planners (like the Depth-Based A* mission planner), they must still be accounted for in real time as other behaviors (i.e., avoiding uncharted obstacles or vessels) might cause the ASV to deviate from the planned path. Using the reactive obstacle avoidance system developed in this work, charted obstacles are accounted for and are avoided in real time without suffering from the challenges of real-time sensor processing. However, to further ensure the ASV's safety, especially around uncharted obstacles, additional real-time obstacle avoidance systems should be implemented that use external sensors. The implementation of these systems has been left for future work.

3.3 Depth-Based A* Mission Planner with Reactive Obstacle Avoidance

The missions shown above clearly illustrate that the Depth-Based A* mission planner and ENC_OA both avoid obstacles in the environment. The recommended usage of this research is to plan the mission with the Depth-Based A* algorithm and to keep the reactive obstacle avoidance system active while navigating. Thus the ASV is provided with a safe mission plan that requires minimal use of the reactive planner, while ensuring the ASV avoids mapped obstacles if the ASV gets off the desired path through other means (i.e., avoiding other boats or environmental factors). One side effect of this recommended usage is that the planned path from the Depth-Based A* algorithm, which may have the ASV navigate within one grid cell away from objects whose depths are seafloor independent, may need to be adjusted to provide an additional margin of safety.

The combination of the two obstacle avoidance methods was tested in simulation near the lateral buoy in the Boston Harbor mission (shown previously in Figure 3.3) using an ASV with

the dimensions of the EchoBoat. In this mission, the planned path was approximately the last kilometer of the mission in Boston Harbor shown earlier in Figure 3.3. As shown in Figure 3.15, the simulated ASV followed the planned path until it was near the lateral buoy. Once the lateral buoy was inside the ENC_Contact's search area, the ASV deviated off of the planned path and drove no closer than 6.5 m from the buffered lateral buoy. Finally, once the ASV was safely beyond the lateral buoy, it drove on the desired path for the remainder of the mission. Without reactive obstacle avoidance, the ASV's planned path would have only been 2.5 m away from the buffered lateral buoy, which may be dangerous in non-ideal conditions (i.e., with wind, current, or waves).

This mission highlights the recommended usage of this work: planning the mission with the Depth-Based A* algorithm and then navigating with ENC_OA (and any other reactive obstacle

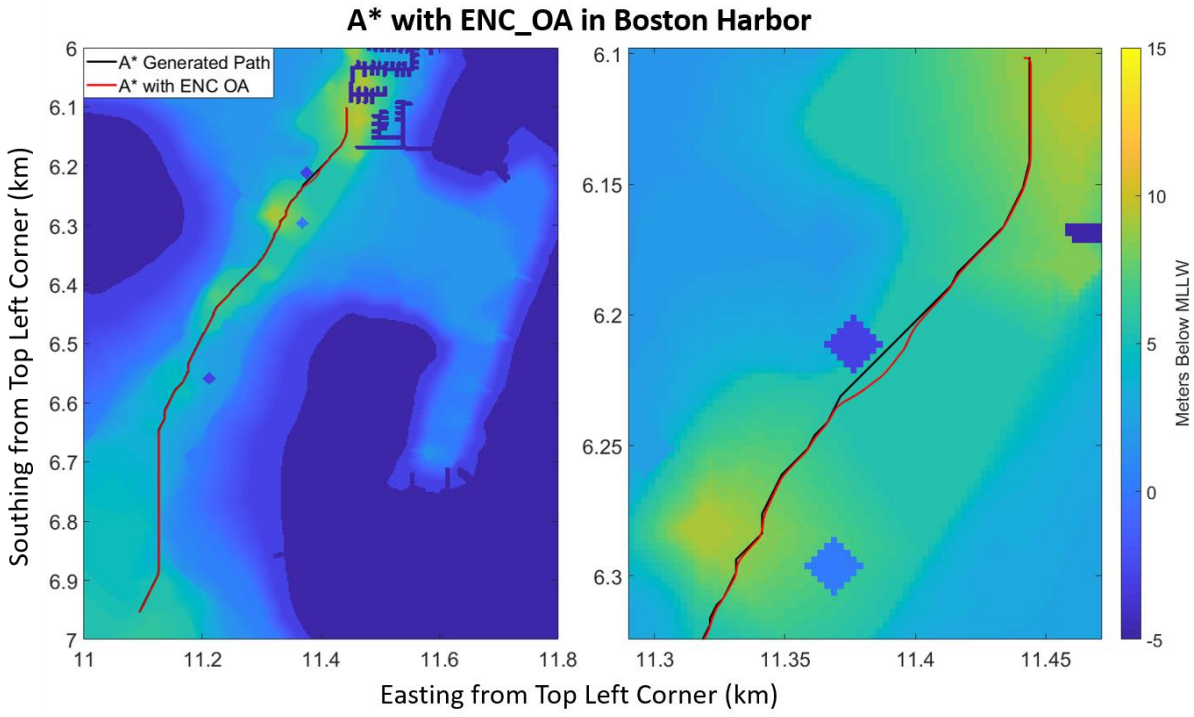


Figure 3.15: Tracklines from a mission planned in Boston Harbor where the red track shows the path traveled by a simulated EchoBoat following a path generated path using the Depth-Based A* Mission Planner (black) with ENC_OA enabled. The left panel gives an overview of the mission and the right zooms in to show where ENC_OA adjusted the planned mission

avoidance not developed by this work) enabled. Thus the path is optimized when possible by using a smart mission planner and ENC_OA increases the robot's safety if it is off the desired track (through environmental factors or otherwise) or, in similar situations to the mission shown in Figure 3.3 and 3.15, where the mission planner drives too close to obstacles for non-ideal conditions.

Chapter 4

Discussion

4.1 Limitations and Future Work

The limitations for the Depth-Based A* mission planner and the reactive obstacle avoidance system are described below along with some workarounds and potential avenues for future work.

4.1.1 Depth-Based A*

The current implementation of the Depth-Based A* mission planner has limitations not addressed in this research. Currently, the mission planner is not optimized for speed, which results from the implementation of the path planner's angular resolution. In the future, the method for increasing the angular resolution could be sped up by decreasing the branching factor or a new, faster method could be defined. Furthermore, Depth-Based A* could be expanded to limit unrealistic changes in heading between waypoints, similar to the method used in [10].

Also, as currently constructed, Depth-Based A* can only run on one chart, so if the start and end points of the mission are not on the same chart, a larger scale chart must be used. The ability to run missions on multiple charts is left for future work. Additionally, the Depth-Based A* mission planner could be expanded to real-time application, as in [14]-[16]. Real-time applications of the mission planner would be especially useful when the ASV's course was adjusted mid-mission from a reactive component, such as those described in this work or one from outside the scope of this thesis (e.g., avoiding boats or other unmapped

obstacles). In a situation where the ASV had diverged significantly from the desired path, a real-time component of the Depth-Based A* mission planner could drive the ASV back on course.

As shown previously in Figure 3.3 and 3.15, the planned path from the Depth-Based A* mission planner may pass within one grid cell from seafloor independent obstacles (e.g., docks, breakwaters, or buoys). Depending on the grid cell resolution and oceanographic conditions, this could represent a potentially dangerous path for the vessel. This limitation can be avoided by enabling the reactive obstacle avoidance also developed in this work. Other solutions are left for future work. Finally, the overall utility of the mission planner is heavily dependent on the chart scale and accuracy, which is described further in Section 4.2. Incorporating additional data sources that augment ENCs is left for future work.

4.1.2 Reactive Obstacle Avoidance

The reactive mission planner outlined in this work successfully avoids obstacles in the ASV's planned path; however, there are some key limitations that an operator must be aware of to maintain the ASV's safety. In the current MOOS build, the helm provides a safe heading for the ASV to navigate, but it does not control the direction in which the ASV can turn. This allows the

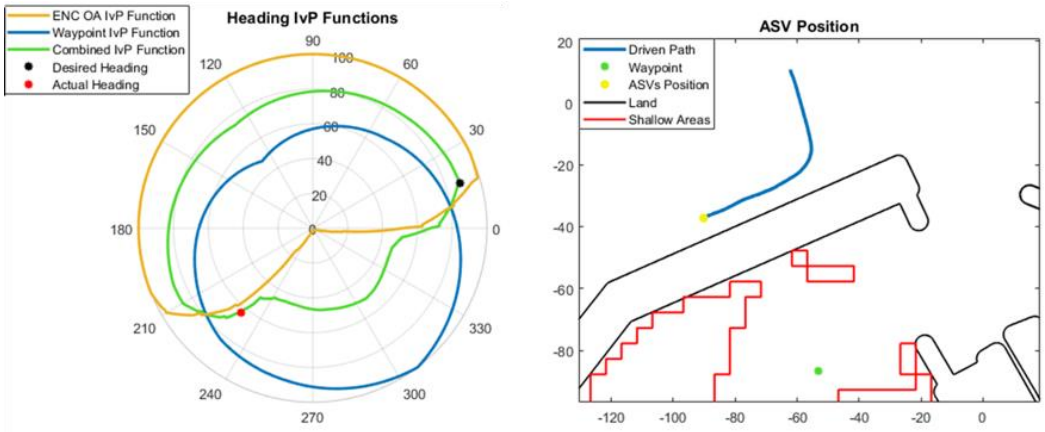


Figure 4.1: Snapshot during the mission where the Echoboat got stuck in a local minimum illustrating how the ASV turns towards obstacles if that is the shortest angular distance between the current and desired heading. The left image shows the heading-based IvP functions and the right image shows a plan view of the mission at the time of the left image

reactive mission planner to change headings towards an obstacle, as shown in Figure 4.1. This situation can become dangerous while navigating in an enclosed space, especially near concave polygons or if the ASV has a large turning radius, as the ASV might run aground. Figure 4.1 illustrates a scenario where the ASV is traveling towards the southwest (red star) and the new desired heading is towards the east (black star). Although turning counterclockwise drives the ASV closer to the pier and through penalized (i.e., dangerous) headings, the ASV turns counterclockwise as that is the shortest angular path to the desired heading. To account for this limitation, the IvP functions for all behaviors (including the default waypoint behavior) must be defined over angular velocity so that it may be controlled by the ASV. Addressing this issue is left for future work.

Additionally, as shown in the field trials, the reactive mission planner can get stuck in local minimum of the IvP Function (see Figures 3.13 and 3.14 where the ASV failed to find a path to drive from one side of the UNH pier to the other). Additionally, there is no guaranteed the path will be the safest when using ENC_OA to avoid obstacles. The mission shown in Figures 3.9 and 3.10 exemplify this behavior. In this mission, the ASV avoided the breakwater near the UNH pier towards the shallow water, south of the pier, rather than towards open water to the north due to a local minimum in the IvP Function. Although the reactive obstacle avoidance system successfully adjusted the ASV's path to avoid all obstacles in the environment, the ASV drove unnecessarily close to both shallow water and the breakwater. Both of these undesirable reactions result from the localized nature of the search area, which is defined as a square centered on the ASV with sides of $20L$ Furthermore when the helm chooses the optimal heading for the ASV, it only considers the heading with the greatest utility during the current iteration, disregarding the previous solution. To

work around this limitation, it is recommended to use the Depth-Based A* mission planner in conjunction with ENC_OA so that the local minima are avoided.

Additionally, as shown with the threat level 5 path on the synthetic obstacle avoidance example (Figure 3.6), the ASV can oscillate in determining which side to avoid the obstacle. This issue shows up when driving at smooth obstacles where the planned path splits the obstacle in half. This indecisive behavior results from the penalty function being heavily dependent on the ASV distance from the obstacle. In the example situation, the northern and southern most extents of the obstacle are approximately the same distance from the ASV and as the ASV turns to avoid the obstacle, it gets closer to the northernmost extent, while moving away from the southernmost extent. This movement penalizes the northernmost extent, which is enough to change the ASV's desired heading and force it to avoid the obstacle towards the south. One potential solution to this problem is to bias the IvP function towards the direction first chosen by the IvP helm. Rectifying this limitation is left for future work.

4.2 ENC Scale and Uncertainty

This work is largely dependent on the accuracy and scale of ENCs. To determine the appropriate resolution for a navigable chart, environmental risk factors and the vessel's physical attributes should be consulted. Hazardous areas (e.g., shoals, breakers, rocky areas, or foul ground) require high levels of positional and depth accuracy so that vessels can make appropriate navigational decisions. In order for small vessels to navigate into places that other ships cannot reach, they need high accuracy positioning equipment. Horizontal positioning from global positioning satellites (GPS) coupled with real time kinematic (RTK) corrections provides centimeter-level precision. Using modern GPS with RTK corrections, the accuracy (and as a result, scale) of the charts becomes the limiting factor.

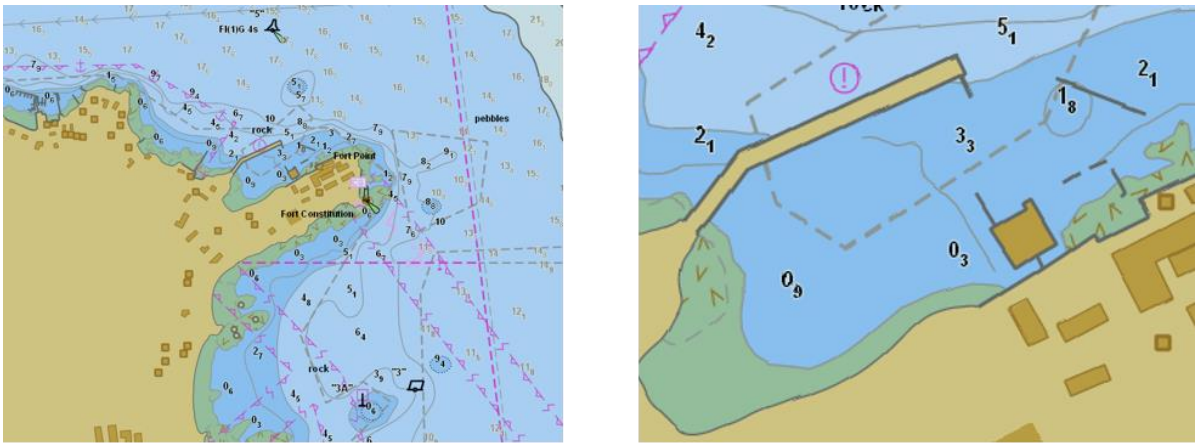


Figure 4.2: On the left, a portion of the Portsmouth Harbor chart is shown at scale, but the chart resolution at this scale is generally inadequate to safely approach the coast. One desires a view closer to the "Navigation scale" representation shown on the right. However in this over-scaled view, features are not uniformly represented at a single scale.

Figure 4.2 illustrates some of the resolution issues with ENCs, where the left image shows a portion of the Portsmouth Harbor ENC (US5NH02) viewed at 1:10,000, which is the highest resolution for this area. However, a chart of this scale is not suitable for navigation in the vicinity of the coastline. When approaching the pier, it is desired to view the chart at a higher resolution, here referred to as the "navigation scale" (shown in the image on the right). However, the chart is now over-scaled. The resulting representation is misleading and can be unsafe for both human and robotic mariners.

The problem lies in attempting to use the chart in a way for which it was not intended. The intended scale of a printed RNC is always clear, as its nature prevents a misleading representation. However, the intended scale of an ENC is less clear as a point has no scale and a line has scale only in a single dimension. Thus, while the annotation of a "chart scale" in the ENC's metadata is helpful and is used in this work, one is too easily tempted to use the power of vector graphics, over-zooming displays or equivalently, producing a grid with an artificially high resolution. On an

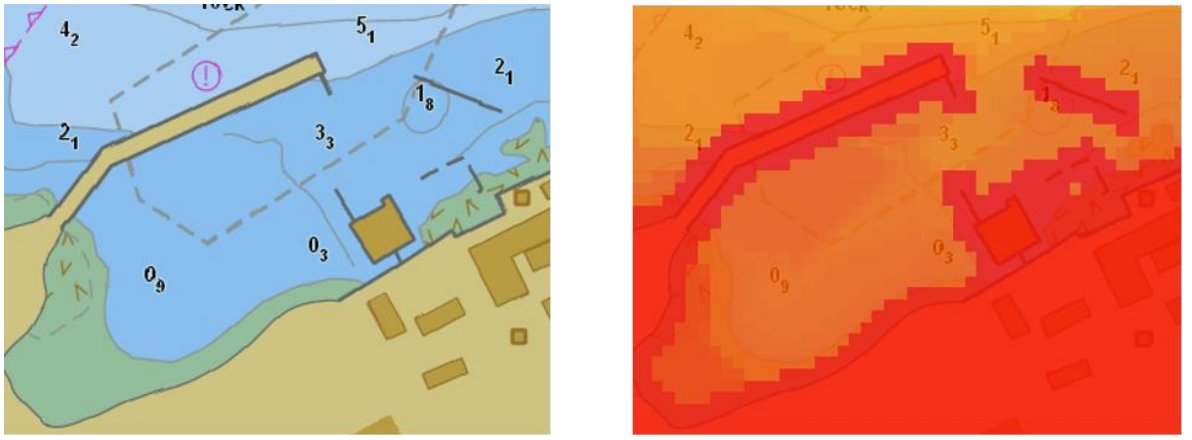
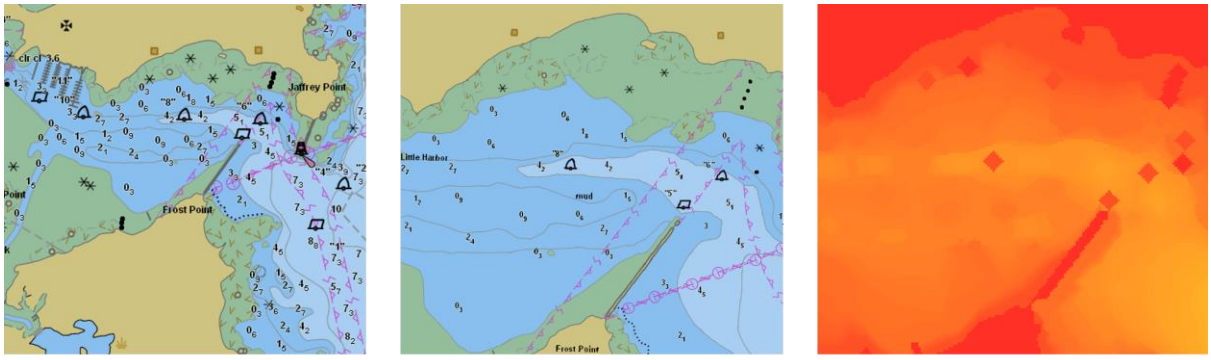


Figure 4.3: The left image properly represents the chart at scale for this area. Thus, we generate a grid for robotic path planning (right) that utilizes the vector representation of the data to retain the proper resolution of features. (Color indicates depth, or risk of grounding)

Electronic Chart Display and Information System (ECDIS) display this is mitigated through the SCAMIN and SCAMAX attributes, which gives the upper and lower limit for the display of the data. However, they are not always available and in this research, a grid resolution of 0.25 mm at chart scale was used. Figure 4.3 depicts the same area of the chart as Figure 4.2 zoomed in to a practical navigational level (left image) and shows this work's representation of the data where color indicates relative risk (i.e., depth) in an attempt to properly represent its underlying fidelity for robotic navigation (right image).

Another example of the dangers of vector graphics is shown in Figure 4.4, where a section of Little Harbor is shown at chart scale (left image), which is inadequate for small vessel navigation. Thus, a higher resolution chart is needed to clearly determine a safe path (middle image). However, now the symbol for rocks, which were previously covered 40 m in diameter when viewed at chart scale, now only cover 10 m. The rock's actual location may be anywhere within that 40 m circle, even though the location of the rock is represented in the ENC as a point with seven decimal precision, which is approximately centimeter-level accuracy. (In some cases,



Little Harbor (Chart Scale) | Little Harbor (Navigation Scale) | Little Harbor (Gridded Cost Map)

Figure 4.4: Little Harbor shown at chart scale (20000:1) as well as at a level conducive to navigation (middle). This work's representation (right) attempts to buffer objects to their original representation at chart scale.

due to the chart compilation process, the actual feature may be outside the 40 m dimeter circle. However, dealing with these outliers are left for future work.) Thus, to safely navigate on the chart, this work buffers all obstacles with point-based geometry so that they are represented as their nominal size at their compilation scale, as shown in the right image of Figure 4.4.

Another challenge in utilizing ENCs for ASV navigation is their completeness. Figure 4.5 illustrates an example of their incompleteness by comparing a satellite image (left) with the

Google Maps | ENC at "Navigation" Scale

Figure 4.5: Satellite image and ENC images of the UNH's Pier Facility. A floating pier exists adjacent to the fixed pier but is not represented in the ENC where only the spatial extent of the fixed pier is shown. Similarly, the breakwater, (installed in 2006) was omitted until a recent update (Summer 2018).

corresponding ENC (right). In this example, only the fixed part of the pier is represented in the ENC. However, the floating pier, which is adjacent to the fixed pier, is only represented in the ENC with a pontoon at the end of the pier instead of subscribing the pier's entire length. Furthermore, this pontoon is represented as a line, when in reality the pontoon has a width of approximately 2.5 m. NOAA's Nautical Charting Manual provides guidance for the representation of multiple piers as a single unit when the compilation scale prevents their clear depiction. [28] Unfortunately, there is no guidance specifying that the resulting depiction should encompass the union of the represented structures. Maps generated from ENCs for safe autonomous navigation must buffer around objects that were not fully depicted in the cartographic representation. In this work, buffering was done using the Equation 2.2 and 2.5 to mitigate this problem.

Figure 4.5 also shows a floating breakwater in the upper right corner of the satellite image. This breakwater was installed in 2006 but remained unrepresented in the chart until a recent update of the ENCs in the summer of 2018. The omission likely resulted from a gap in coverage between those who delineate coastline and those who map coastal hazards. This omission, although recently fixed, epitomizes some of the challenges faced by utilizing only ENCs for the source data.

Finally, Figure 4.6 presents a long-known issue with cartographic representations that is quite challenging for robotic systems: white water that mariners should avoid indicated by the word "Breakers" on the RNC. The satellite image overlain on the RNC (left image) shows that the actual breakers are approximately 88 m from the represented location. The right image shows the ENC overlain on a bathymetric grid. No shoal area exists where the breakers are indicated. The bathymetry further suggests that the white water in the satellite image is the correct location. It looks as if there was simply insufficient space on the raster chart for the word "Breakers" where the breakers actually exist. In the associated ENC, the reported location of "water turbulence"

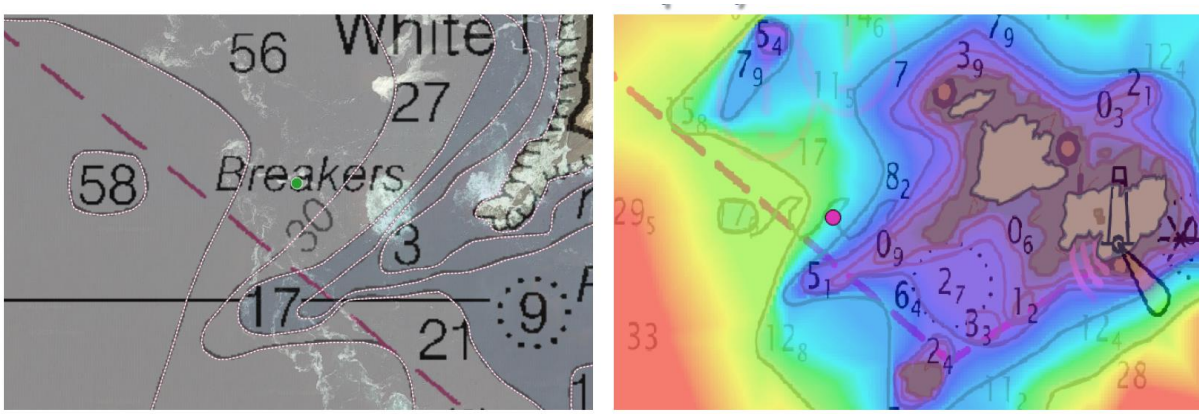


Figure 4.6: The raster chart with satellite image overlay (right) shows the word "Breakers" indicating hazardous water conditions. However, the location of the actual hazard is 88 m away from the actual hazard, which is apparent as white water in the satellite image, and confirmed by the underlying bathymetry in the right image. The location is misrepresented due to the cartographic depiction.

(shown with the green and red dots on the left and right images respectfully) are precisely where the "Breakers" had been on the raster predecessor.

The cartographic nuance illustrated by the "Breakers" example in Figure 4.6 is difficult for robotic vessels to interpret from the data provided in the ENC. The utility of a coastal chart is all but lost if one must buffer all objects conservatively enough to allow for archaic cartographic discrepancies resulting from the translation of raster-based charts. Without more accurate information, satellite imagery or other sources of data would be required to confirm the proper location of hazards for safe ASV work.

To maintain the safety of both mariners and ASVs, a dual position encoding scheme for objects in ENCs should be put into place. The first value should be the actual position of the hazard as well as it can be measured and the second should provide an offset from that position for the recommended display location. Using this scheme, the fidelity of the data would be maintained as the true accuracy of the depth and position measurement would remain, while at the same time

provide guidance for human mariners (and ASV operators) on the location of object's representation on the chart.

Additionally, objects that have depth should always be reported with a quantitative depth measurement and not just a qualitative depth assessment through the water level attribute. Although this might not be attainable for all obstacles whose depths are inside the tidal cycle, we feel that all obstacles that remains underwater through the tidal cycle (WATLEV = 3) should have their depth recorded quantitatively.

As the methods in this work relies on the validity of ENCs, it is vital for charts to be accurate, but charts age discrepancies between charted depth and true depths may increase. Furthermore, in some locations, such as some coastal Alaskan waters, high resolution charts simply do not exist. In these locations, the compilation scale, which are often on the order of 1:120,000, may not be sufficient for near shore navigation. In these situations, this author believes that additional sensors are necessary to ensure the ASV's safety. However, charted hazards should not be disregarded entirely and the methods described in this thesis are still applicable, even in these worse case scenarios, for path planning between points of interest, as charts give the ASV NOAA's best published *a priori* information on its environment.

Chapter 5

Conclusion

A complete package of obstacle avoidance using ENCs was developed in this research with both an *a priori* mission planner and a reactive obstacle avoidance system to help ASVs safely navigate their environment. This system allows a COTS ASV, like the Seafloor Systems EchoBoat or the ASV Global C-Worker 4, to increase its autonomy by giving it an understanding of its environment using ENCs. ENCs give information on a wide range of potential obstacles from coastlines to underwater rocks to buoys and breakwaters. The information is easily accessed and does not require sophisticated onboard sensor processing to avoid known features.

Like [14]-[16], this research uses a hybrid approach that incorporates both a mission planner and reactive obstacle avoidance system, as when the two methods are combined, the flaws of each system are minimized. The Depth-Based A* mission planner uses a novel cost function which simulates the actions of a human mariner by prioritizing driving in the channel over shoal areas. This ability was made possible by the interpolation of the ENC into a depth-based cost map. Additionally, the reactive obstacle avoidance behavior successfully showed in both simulation and the field the ability to avoid obstacles with complex geometries.

Human derived, static mission plans severely limits the feasibility of ASVs. Without an intelligent mission plan with reactive obstacle avoidance, a human operator must supervise the ASV and account for all hazards in its environment. Providing an ASV the ability to read, understand, and utilize nautical charts allows for the safe avoidance of known obstacles in the environment. No mariner goes into an unfamiliar harbor without consulting a nautical chart. Autonomous surface vehicles should not be an exception.

References

[1] Department of the Navy. (2007 July 23). “The Navy Unmanned Surface Vehicle (USV) Master Plan.” [Online] Available: http://www.navy.mil/navydata/technology/usvmppr.pdf

[2] E. Oleynikova, N. B. Lee, A. J. Barry, J. Holler, and D. Barrett, “Perimeter patrol on autonomous surface vehicles using marine radar,” in *OCEANS 2010 IEEE - Sydney*, 2010, pp. 1–5.

[3] D. Manda, “Development of autonomous surface vessels for hydrographic survey applications,” University of New Hampshire, Durham, New Hampshire, 2016.

[4] H. Ferreira, A. Martins, J. M. Almeida, C. Almeida, A. Dias, E. Silva, and G. Silva, “Environmental modeling with precision navigation using ROAZ autonomous surface vehicle,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots

[5] J. E. Manley, “Unmanned surface vehicles, 15 years of development,” in OCEANS 2008, 2008, pp. 1–4.

[6] M. Caccia et al., “Sampling sea surfaces with SESAMO: an autonomous craft for the study of sea-air interactions,” IEEE Robotics Automation Magazine, vol. 12, no. 3, pp. 95–105, Sep. 2005.

[7] M. Dunbabin, A. Grinham, and J. Udy, “An autonomous surface vehicle for water quality monitoring,” in Australasian Conference on Robotics and Automation (ACRA), Sydney, Australia, 2009.

[8] H.-M. Huang, “Autonomy Levels for Unmanned Systems (ALFUS) Framework: Safety and Application Issues,” in Proceedings of the 2007 Workshop on Performance Metrics for Intelligent Systems, New York, NY, USA, 2007, pp. 48–53.

[9] IHO Transfer Standard for Digital Hydrographic Data, Special Publication No. 57 3rd Edition, November 2000.

[10] J.-M. Yang, C.-M. Tseng, and P. S. Tseng, “Path planning on satellite images for unmanned surface vehicles,” International Journal of Naval Architecture and Ocean Engineering, vol. 7, no. 1, pp. 87–99, Jan. 2015.

[11] P. E. Hart, N. J. Nilsson, and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[12] K. Naus and M. Wąż, “The idea of using the A* algorithm for route planning an unmanned vehicle ‘Edredon,’” Zeszyty Naukowe / Akademia Morska w Szczecinie, no. 36 (108) z. 2, pp. 143-147, 2013.

[13] C. Sauze and M. Neak, “A Raycast Approach to Collision Avoidance in Sailing Robots,” in International Robotic Sailing Conference, 2010, pp. 25–32.

[14] J. Larson, M. Bruch, and J. Ebken, “Autonomous Navigation and Obstacle Avoidance for Unmanned Surface Vehicles,” in Proc. SPIE Unmanned Systems Technology VIII, 2006, vol. 6230.

[15] J. Larson, M. Bruch, R. Halterman, J. Rogers, and R. Webster, “Advances in Autonomous Obstacle Avoidance for Unmanned Surface Vehicles,” AUVSI Unmanned Systems North America 2007, 2007.

[16] G. Casalino, A. Turetta, and E. Simetti, “A Three-Layered Architecture for Real Time Path Planning and Obstacle Avoidance for Surveillance USVs Operating in Harbour Fields,” in OCEANS 2009 - EUROPE, 2009, pp. 1–8.

[17] E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs,” Numer. Math., vol. 1, no. 1, pp. 269–271, Dec. 1959.

[18] M. Benjamin, H. Schmidt, and P. Newman. (2017, July) An Overview of MOOSIvP and a User’s Guide to the IvP Helm - Release 17.7. MIT. [Online]. Available: http://oceanai.mit.edu/ivpman/pmwiki/pmwiki.php

[19] P. Newman, “MOOS — a mission oriented operating suite,” Department of Ocean Engineering, Massachusetts Institute of Technology, Tech. Rep., 2002.

[20] M. R. Benjamin, “The Interval Programming Model for Multi-Objective Decision Making,” Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, MA, Tech. Rep. AIM-2004-021, September 2004.

[21] GDAL. 2018. GDAL - Geospatial Data Abstraction Library: Version 2.3.2, Open Source Geospatial Foundation, http://gdal.osgeo.org

[22] National Oceanic and Atmospheric Administration. (April 2018) “Hydrographic Surveys Specifications and Deliverables.” [Online] Available:

https://nauticalcharts.noaa.gov/publications/docs/standards-and-requirements/specs/hssd-2018.pdf

[23] ASV Global. C-worker 4. ASV Global. [Online]. Available: https://www.asvglobal.com/wp-content/uploads/2018/06/C-Worker-4_Datasheet_2018.pdf

[24] S. Cox. (2013). Pytides v0.0.4 [Online]. Available: https://github.com/sam-cox/pytides

[25] M. G. G. Foreman and R. F. Henry, "The harmonic analysis of tidal model time series," Advances in Water Resources, vol. 12, no. 3, pp. 109–120, Sep. 1989.

[26] Seafloor Systems. EchoBoat-ASV. [Online] Available: https://seafloorsystems.com/support/brochures/seafloor-docs/129-echoboat-asv-autonomous-surface-vehicle-brochure/file

[27] National Imagery and Mapping Agency. "Chapter 14 Electronic Charts." [Online] Available: http://msi.nga.mil/MSISiteContent/StaticFiles/NAV_PUBS/APN/Chapt-14.pdf

[28] "Nautical Chart Manual Volume I, Policies and Procedures" Version 2016.2, U.S. Department of Commerce - Office of Coast Survey, 2/19/2016.