PUCRS

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

DARLAN ALVES JURAK

A COLREGS-COMPLIANT COLLISION AVOIDANCE SYSTEM FOR UNMANNED SURFACE VEHICLES

Porto Alegre
2020

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica do Rio Grande do Sul

PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
SCHOOL OF TECHNOLOGY
COMPUTER SCIENCE GRADUATE PROGRAM

A COLREGS-COMPLIANT COLLISION AVOIDANCE SYSTEM FOR UNMANNED SURFACE VEHICLES

DARLAN ALVES JURAK

Thesis submitted to the Pontifical Catholic University of Rio Grande do Sul in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Prof. Dr. Alexandre de Morais Amory
Co-Advisor: Dr. Vitor Augusto Machado Jorge

Porto Alegre
2020

Dados Internacionais de Catalogação na Publicação (CIP)

J95c Jurak, Darlan Alves

A COLREGS-compliant collision avoidance system for unmanned surface vehicles / Darlan Alves Jurak. – 2020.

70 p.

Dissertação (Mestrado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientador: Prof. Dr. Alexandre de Morais Amory.

Co-orientador: Dr. Vitor Augusto Machado Jorge.

1. Unmanned Surface Vehicle. 2. COLREGS. 3. A*. 4. ROS. I. Morais Amory, Alexandre de. II. Jorge, Vitor Augusto Machado. III. Título.

Darlan Alves Jurak

A COLREGS-COMPLIANT COLLISION AVOIDANCE SYSTEM FOR UNMANNED SURFACE VEHICLES

This Master Thesis has been submitted in partial fulfillment of the requirements for the degree of Master of Computer Science, of the Graduate Program in Computer Science, School of Technology of the Pontifícia Universidade Católica do Rio Grande do Sul.

Sanctioned on March 27th, 2020.

COMMITTEE MEMBERS:

Prof. Dr. Edson Prestes (UFRGS)

Prof. Dr. Felipe Rech Meneguzzi (PUCRS)

Prof. Dr. Alexandre de Morais Amory (PPGCC/PUCRS - Advisor)

Dr. Vitor Augusto Machado Jorge (Co-Advisor)

Dedico este trabalho principalemente a minha mãe, Léia e a minha avó, Venina, sem elas nada seria possível. Sempre me incentivaram a ir em busca dos meus sonhos, me fizeram acreditar que posso sempre alcançar o que desejo e me deram suporte imensurável. Dedico também aos meus amigos, Marcos, Vini, Duka, Fran, Zé, Bruno, Lima, Bruna, Olimar e Mateus; dindos e colegas, sem o apoio, carinho e momentos de diversão a trajetória seria muito difícil. Dedico este trabalho especialmente a minha namorada, Gabi, ela foi quem aguentou minhas ausências em dias de mais de 14h/dia de dedicação a este trabalho e muitos compromissos desmarcados.

"If I have seen further than others, it is by standing upon the shoulders of giants."
(Isaac Newton)

UM SISTEMA DE EVASÃO DE COLISÃO COM RESPEITO À COLREGS PARA VEÍCULOS NÃO-TRIPULADOS QUE NAVEGAM NA SUPERFÍCIE DA ÁGUA

RESUMO

Os veículos de superfície não tripulados (USVs) constituem uma categoria de robôs aquáticos que atuam sem tripulação na superfície da água, apresentando comportamento autônomo ou sendo controlados remotamente. Nas últimas décadas, vários estudos foram realizados para tornar os USVs autônomos. As atuais aplicações de USVs incluem monitoramento do ambiente, exploração de recursos oceânicos como petróleo e gás, vigilância portuária e costeira para fins militares, transporte e pesquisa científica. O Regulamento Internacional para evitar Abalroamento no Mar (RIPEAM) determina regras que devem ser seguidas por marinheiros para evitar colisões em possíveis cenários de colisão, como cruzamento, encontro frontal e ultrapassagem. Atualmente, colisões diretas entre navios representam 60% dos acidentes no mar, e 56% das colisões são causadas por violação do RIPEAM. Portanto, os USVs devem estar em conformidade com o RIPEAM. Nesta dissertação, apresentamos o sistema que desenvolvemos para orientar USVs em missões autônomas e em conformidade com o RIPEAM. O principal módulo de orientação do nosso sistema é um planejador de caminhos que respeita o RIPEAM. Adaptamos a técnica de Artificial Terrain Cost, apresentado por Agrawal [3] *et al.*, à nossa solução baseada em A*, dessa forma, bloqueamos no espaço de busca as posições não compatível com o RIPEAM, através da criação de obstáculos virtuais. Desenvolvemos o nosso sistema usando o ROS (Robotic Operating System) e integramos ele a um barco diferencial disponível no simulador USV_sim, um simulador para USVs, capaz de gerar distúrbios realistas, como vento e corrente de água. Para a avaliar o sistema proposto, medimos o tempo computacional, o vento máximo sustentado e a distância mínima mantida ao encontrar outra embarcação nos cenários de encontro frontal, de cruzamento pela esquerda, cruzamento pela direita e ultrapassagem.

Palavras-Chave: Unmanned Surface Vehicle (USV), Normas da Autoridade Marítima (NORMAM), Evasão de Colisão, Artificial Terrain Cost A*, Robotic Operating System (ROS).

A COLREGS-COMPLIANT COLLISION AVOIDANCE SYSTEM FOR UNMANNED SURFACE VEHICLES

ABSTRACT

Unmanned surface vehicles (USVs) are a category of aquatic robots that act unmanned on the water surface, performing autonomous behavior or being remotely controlled. In recent decades, several studies have been carried out to make USVs autonomous. The main challenges are related to collision avoidance, accurate navigation on the high seas, and compliance with international maritime rules, such as the Convention on the International Regulations for Preventing Collisions at Sea (COLREGS). Current applications of USVs include monitoring the environment, exploiting ocean resources such as oil and gas, port and coastal surveillance for military purposes, transport, and scientific research. COLREGS determines rules that must be followed by sailors to avoid collisions in possible collision scenarios, such as crossing, head-on, and overtaking. Currently, direct collisions between vessels represent 60% of accidents at sea, and the violation of COLREGS causes 56% of collisions. Therefore, USVs must comply with COLREGS. In this master's thesis, we present the system we developed to guide USVs on autonomous missions while following the COLREGS. The main guidance module of our system is a path planner that follows the COLREGS. We adapted the Artificial Terrain Cost method, presented by Agrawal [3] *et al.*, to our solution, blocking locations in the search space that are not compliant with COLREGS, through the creation of virtual obstacles. We integrated our system to a differential boat available in the USV_sim simulator, a simulator for USVs, capable of generating realistic disturbances, such as wind and water current. To evaluate the proposed system, we measure the computational time, the maximum sustained wind, and the minimum distance maintained when encountering another vessel in the scenarios of head-on, crossing from the left, crossing from the right, and overtaking encounters.

Keywords: Unmanned Surface Vehicle (USV), Convention on the International Regulations for Preventing Collisions at Sea, Collision Avoidance (COLREGS), Artificial Terrain Cost A*, Robotic Operating System (ROS).

LIST OF ACRONYMS

AIS Automatic Identification System

AMCL Adaptative Monte Carlo Localization

ARPA Automatic Radar Plotting Aid

ATC Artificial Terrain Cost

AV Approaching Vessel

COA Circle of Acceptance

COR Circle of Rejection

CPA Closest Point of Approach

COLREGS Convention on the International Regulations for Preventing Collisions at Sea

DGPS Differential Global Positioning System

DPSS Direction Priority Sequential Selection

GNC Guidance, Navigation and Control

GP Genetic Programming

GPS Global Positioning System

GUI Graphical User Interface

HTN Hierarchical Task Network

IMO International Marine Organization

IMU Inertial Mesurement Unit

JSHOP Java Simple Hierarchical Ordered Planner

LADAR Laser Radar

LIDAR Light Detection and Ranging

LOS Line of Sight

LSA Autonomous System Laboratory

MPC Model Predictive Control

MPSoC Multi-Processor System-on-Chip

MVFF Modified Virtual Force Field

MWR Milimeter Wave Radar

NORMAM *Normas da Autoridade Marítima*

ODE Ordinary Differential Equation

OV Own Vessel

PEP Research Plan

PID Proportional–Integral–Derivative

POC Proof-of-Concept

RGBD Red-Green-Blue-Depth

RIPEAM Regulamento Internacional para evitar Abalroamento no Mar

ROS Robotic Operating System

R-RA* Rule-based Repairing A*

SA *Seminário de Andamento*

SONAR Sound Navigation and Ranging

TSS Traffic Separation Schemes

USV Unmanned Surface Vehicle

VFF Virtual Force Field

VO Velocity Obstacle

CONTENTS

1 INTRODUCTION **12**
1.1 RESEARCH PROBLEM AND SCOPE 12
1.2 CONTRIBUTIONS 13
1.3 PUBLICATIONS 13
1.4 THESIS OUTLINE 14

2 THEORETICAL BACKGROUND **15**
2.1 GNC SYSTEM 15
2.2 COLREGS 19

3 LITERATURE REVIEW **21**
3.1 LITERATURE REVIEW 21
3.2 LITERATURE DISCUSSION 32

4 COLREGS-COMPLIANT SYSTEM **35**
4.1 ASSUMPTIONS AND LIMITATIONS 35
4.2 SYSTEM ARCHITECTURE 35

5 SIMULATION AND RESULTS **44**
5.1 SIMULATIONS CHARACTERIZATION 44
5.2 EXPERIMENTS RESULTS 47
5.3 HEAD-ON 47
5.4 CROSSING FROM THE RIGHT 51
5.5 CROSSING FROM THE LEFT 54
5.6 OVERTAKING 57
5.7 RESULTS SUMMARY 60

6 CONCLUSION **63**
6.1 RESULTS DISCUSSION 63
6.2 FUTURE WORK 64

REFERENCES **66**

1. INTRODUCTION

Approximately 71% of the Earth is covered by water, and essential tasks happen on its surface, such as environment monitoring, merchandising, and exploration. Some of these tasks can be dangerous, exhaustive, or tedious for humans, so a trend is the development of autonomous systems for executing these tasks. Thus, driven by military, scientific, and commercial interests, the development of Unmanned Surface Vehicles (USVs) has become a current demand [29].

USVs constitute a category of aquatic robots that act without a crew on the water surface, presenting autonomous behavior or being remotely controlled. In the last decades, several techniques have been applied for making USVs autonomous. The main challenges are related to collision avoidance, precise navigation on the seagoing, and accordance with international marine rules, namely Convention on the International Regulations for Preventing Collisions at Sea (COLREGS). Current USV applications include: environment monitoring [9]; ocean resources exploration as oil and gas exploration [37]; port, harbor and coastal surveillance for military purposes [8, 37, 45]; transportation [24]; and scientific research [22].

The COLREGS determines rules that must be followed by maritime pilots for preventing collisions in potential collision scenarios such as crossing, head-on, and overtaking. Currently, direct collision between ships represents 60% of the accidents at sea, and 56% of the collisions are caused by COLREGS violation [29, 11]. Thus, USVs must be compliant with the COLREGS.

1.1 Research Problem and Scope

USV COLREGS compliant systems already exist and are being developed extensively. We note that many works on USV have the proposal of developing specific applications with USV and sometimes end up not providing access to their applications, or are protected with patents. In our laboratory, we have small unmanned vessels with the potential to be used in disaster situations during the monitoring phase or in activities to collect water samples for quality assessment. The focus of this work is to generate a collision avoidance system that can serve as a basis for the autonomous guidance of our vessels. As respect for the international rules of the navy is of high relevance, we chose to develop the base system with COLREGS compliance.

Our scope is in simulation, we developed and validated our system using the USV_sim[1] [35, 36] simulator. The USV_sim is a simulator developed by our research group, which besides allowing simulation considering realistic disturbances such as wind influence,

[1] https://github.com/disaster-robotics-proalertas/usv_sim_lsa

also has realistic representations of the vessels that we have in our research group. For the development of our system, we use the Robotic Operating System (ROS) [39] framework that is widely used around the world, and we believe this way, we can continue to develop our system in a distributed and collaborative way.

1.2 Contributions

The main contribution of this work is the integration of guidance, navigation, and control modules to compose a COLREGS-compliant system. The strategy used for the system to be COLREGS-compliant consisted of replicating the collision avoidance method presented by Agrawal [3] and integrating it into the guidance module. Therefore, in this work we implemented a path planner composed of A * and virtual obstacles [3]. Many modules that compose our guidance, navigation, and the control systems are nothing new, but to the best of our knowledge, the integration of modules for the composition of a single system capable of controlling a USV is a relevant contribution to the scientific community. In this way, the contribution of the work is also to provide a navigation framework for USV.

As secondary contribution we developed our system under the ROS framework to enable collaborative and distributed development. Also, we make the proposed system available as a ROS plugin[2]. Furthermore, regarding the evaluation of our system, we integrated it into the USV_sim simulator. This integration allow us and future users to evaluate and use our system considering realistic environmental disturbances such as wind, water current and waves influence. Our system can be used as a base system for further development.

1.3 Publications

During the master's period, the author has a journal article and two papers accepted for publication, presented as follows.

1. 2019
    - Journal article: "A Survey on Unmanned Surface Vehicles for Disaster Robotics: Main Challenges and Directions" [23]. This paper presents the first comprehensive survey, to the best of our knowledge, about the applications and roles of USVs for disaster management. Currently, we have 11 citations around the world.
    - Symposium article: "Programming teaching with robotic support for people who are visually impaired: a systematic review" [13]. In our laboratory, we developed a

[2]https://github.com/Unmanned-Surface-Vehicle/atc_astar

robotic environment composed of programming language, simulation, and a robot to aid people who are visually impaired to learn programming. In this paper, we extended a review of other works that aid visually impaired people on learning programming with robotics.

- Symposium article: "Integrating an MPSoC to a Robotics Environment [14]. In this work, we integrated a Multi-Processor System-on-Chip (MPSoC) and a robotic simulator and ran a trivial application for demonstration purposes.

1.4 Thesis Outline

In Chapter 2, we present important definitions and background information related to our research. In Chapter 3, we present and discuss the literature related to USVs guidance systems. In Chapter 4, we present the developed system, its architecture and features. In Chapter 5, we present simulation scenarios and results. In Chapter 6, we discuss the collected results.

2. THEORETICAL BACKGROUND

In this chapter, we present the background of the main concepts regarding our study. The focus is to present the main software components of an Unmanned Surface Vehicle (Section 2.1), and present the international regulations (COLREGS) for navigation of vessels on water (Section 2.2).

2.1 GNC System

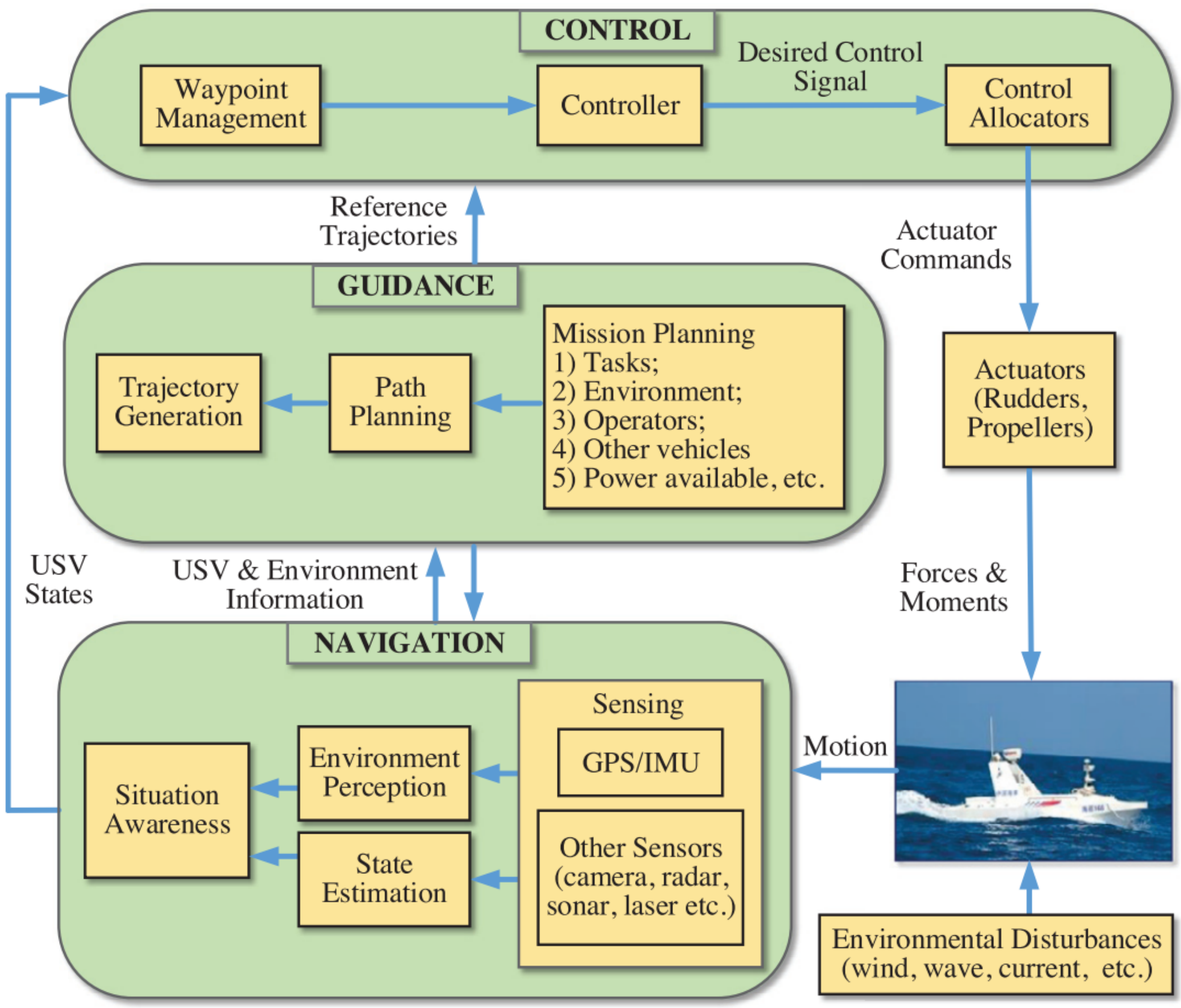


Figure 2.1: GNC System Modules [29]

The GNC system is an essential component for most USVs, being responsible for managing partially or entirely the USV. GNC stands for Guidance, Navigation, and Control. In this section, we describe the responsibilities of each one of the GNC modules as well as the interaction between them. The Figure 2.1, presented by Liu [29], summarizes the GNC modules, their responsibilities, and interactions.

In short, the guidance system is responsible for the determination of the USV's trajectory to achieve a goal location. For trajectory generation, the guidance system uses the information provided by the navigation system regarding the environment (*e.g.,* obstacles around and general disturbances such as wind and current) and related to the USV's state (*e.g.,* current location). After generating the trajectory, the guidance system makes the route available to the control system. The control system generates actuation commands to change the USV's state and effectively move the USV. We present detailed explanations about the GNC modules in the following sections.

2.1.1 Guidance System

In an autonomous approach, most USVs guidance systems are responsible for planning the path that will be traveled by the USV. For the determination of a path, the guidance system uses the information gathered by the navigation system regarding the environment and the USV's state. In general, trajectory generation shall consider the USV mission and marine protocols, such as being under the COLREGS or respect Traffic Separation Schemes (TSS) definitions (TSS are similar to traffic ground lines that must be respected by vessels on navigation at sea.) Also, information about vehicle capability (e.g., maximum speed and power consumption) and environmental conditions (e.g., wind, wave, and current disturbance) may be required to determine suitable trajectories.

Typical implementations of the guidance system propose the usage of global and local planners [29]. The global planner is responsible for path planning related to the far-field based on well-known information about the environment, such as islands, coasts, bridges over water *etc.*, assuming a deliberative behavior. Conversely, the local planner is responsible for path planning regarding the near-field environment, assuming a reactive behavior when detecting other vessels or obstacles. Conventional methods for path planning applied to USV guidance are based on heuristic search [29], such as A* [26, 34]. Some works used optimization methods for USV‘s path planning but presented several limitations in real-world trials, due to expensive computational cost [47, 10].

In Figure 2.2, we illustrate the difference between global and local planning in our context. For a situation where our vessel (henceforth referred to as Own Vessel (OV)) is not capable of detecting some vessel in the far-field, the global planner could define a route above a location occupied by an approaching vessel (henceforth referred to as Approaching Vessel (AV)). In this situation, the local planner should react when an AV appears in the near-field and define an avoidance behavior. Both global and local planners usually define the USV trajectory considering static and dynamic obstacles.

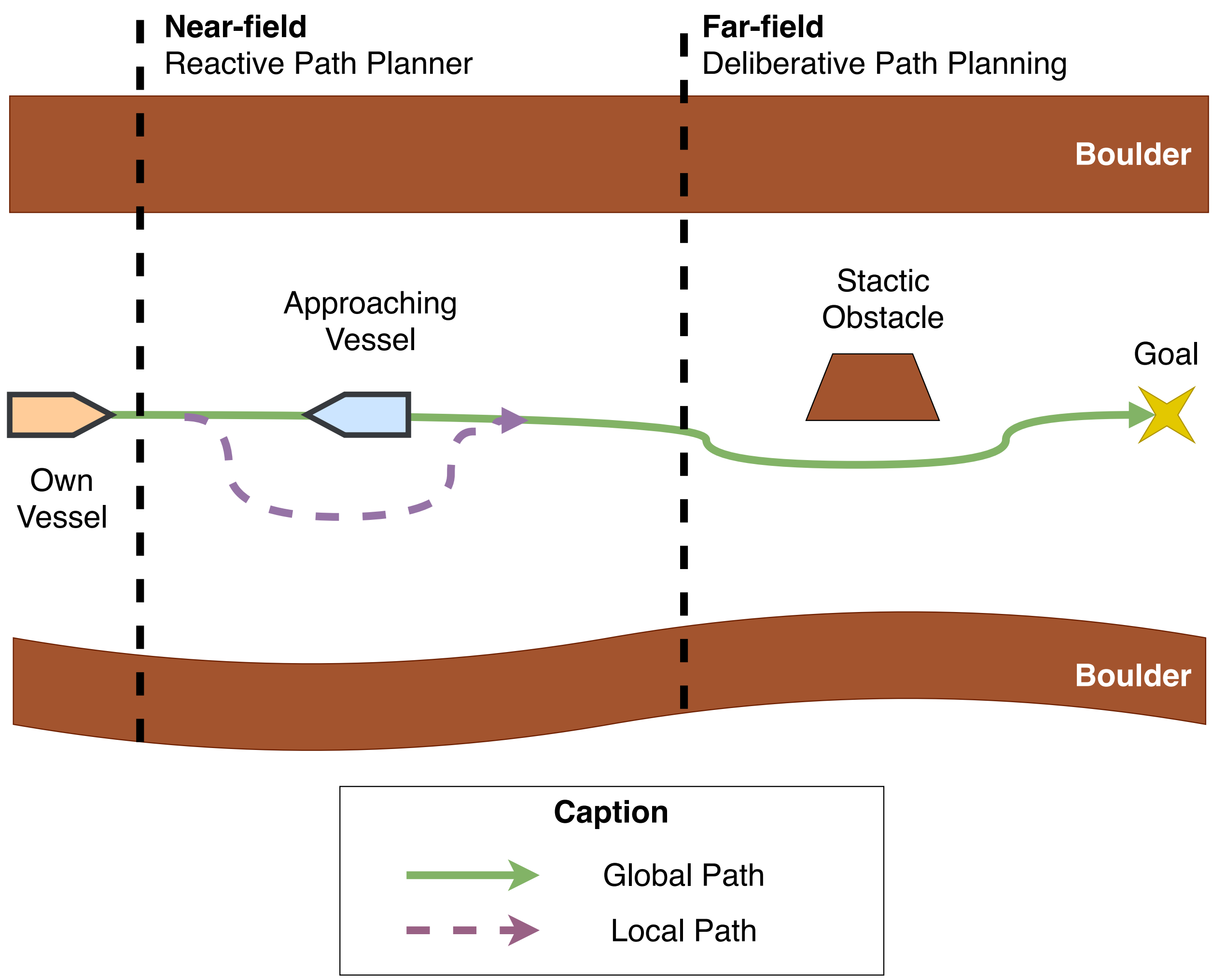


Figure 2.2: Global and local paths.

In general, static information about the environment, such as islands, and coasts, are extracted from nautical charts, topography charts, and offline available maps. While dynamic information about the environment, such as unknown static obstacles and other vessels, is acquired in run-time by the navigation system. Also, the guidance system depends on a world representation (i.e., cost maps) for running its main component, the path planner.

2.1.2 Navigation System

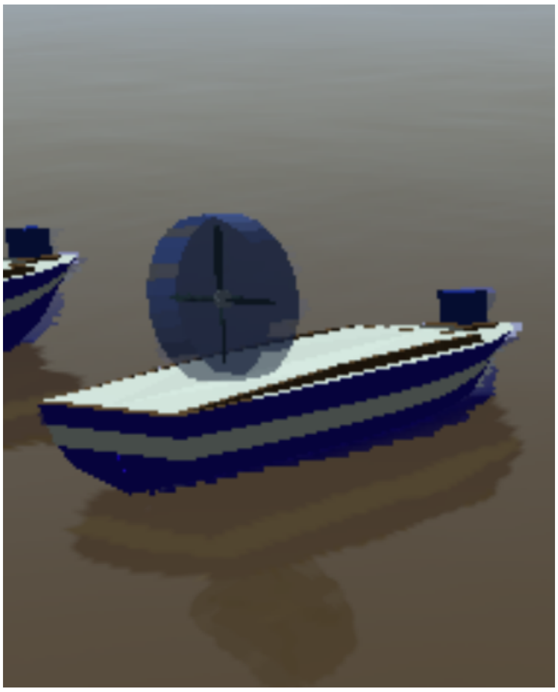
The navigation system is responsible for the determination of the current state of the USV (e.g., position, velocity, orientation, and acceleration) and environment perception (e.g., static and dynamic obstacles, other vessels, wind speed, and water current). For USV's state estimation, the navigation system can use GPS, Inertial Mesurement Unit (IMU)[1], and compasses. Regarding environment perception, static information about

[1] For more information read https://en.wikipedia.org/wiki/Inertial_measurement_unit

the environment is extracted from nautical charts and topographic maps. While, dynamic information about the environment is acquired in real-time sensing by the navigation system through the usage of cameras, Light Detection and Ranging (LIDAR)[2], radar, and Automatic Identification System (AIS). AIS allows automated message exchange between vessels, facilitating the identification of location, velocity, course, path, dimension, and type of AVs.

2.1.3 Control System

The control system is responsible for the generation of actuation commands that change the USV's state. For an airboat (as shown in Figure 2.3a), for example, the control system will generate commands to change the rotation and gain of the fan. While for a differential boat (as shown in Figure 2.3b), the control system will generate a gain command for each thruster. The control system determines actuation commands, considering the trajectory generated by the guidance system.



(a) Airboat with fan

(b) Differential boat with two thruster

Figure 2.3: Example of vessels and different propellers. Both vessels are simulated versions of Platypus [30] boats we have in our laboratory.

[2]For more information read https://en.wikipedia.org/wiki/Lidar

2.2 COLREGS

COLREGS[21] stands for "International Regulation for Preventing Collisions at Sea, 1972", sometimes cited as "COLlision REGulations at Sea" or "Convention on the International Regulations for Preventing Collisions at Sea" - in this work we adopted the latter. The definitions declared on the COLREGS are controlled, updated, and of responsibility of the International Marine Organization (IMO). Briefly, the COLREGS defines the rules that must be followed by vessels upon waters to avoid collisions when encountering another vessel. The COLREGS are adopted by the United Nations as a global convention and must be respected by every country.

COLREGS rules were not defined considering autonomous systems such as USVs. They were written to be interpreted by well-experienced sailors and imply the usage of their experience and common sense. There are gaps to be filled and subjective or ambiguous definitions to be addressed, making the development of a COLREGS-compliant USV guidance system challenging.

Below we describe the four main encounters presented in COLREGS (illustrated in Figure 2.4), head-on, crossing from the right, crossing from the left, and overtaking.

- Head-On: In this situation both vessels should avoid collision going to their starboard side.

- Crossing from the right and crossing from the left: In this scenario, the COLREGS defines different responsibilities for each vessel. The vessel that has the other vessel in its starboard side must give way and is responsible for collision avoidance. The vessel that has the other vessel in its port side must keep its course without significant changes. In a critical situation, where the give-way vessel seems not able to avoid the collision, the keep vessel should act to avoid.

- Overtake: In this scenario the vessel overtaking another, can freely decide which side to go and must avoid generating another crossing situation.

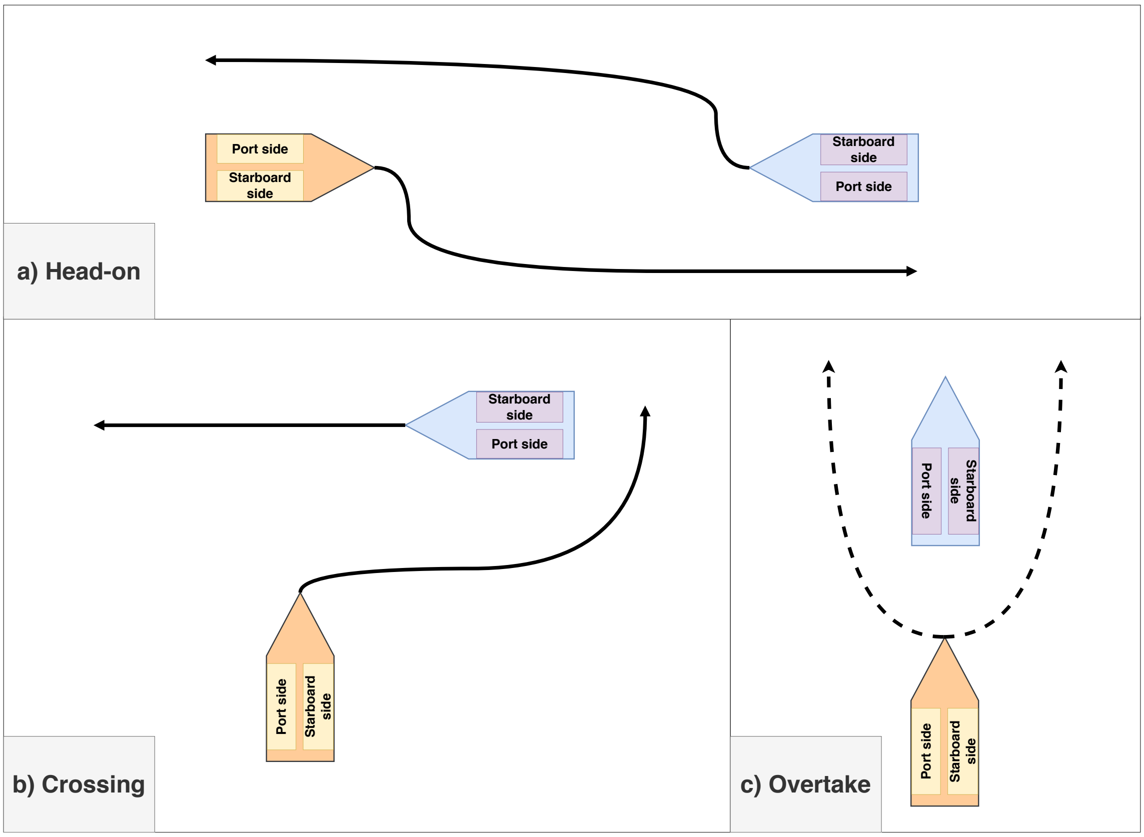


Figure 2.4: Possible encounters between two vessels discussed in the COLREGS. Arrows indicate the behavior that must be performed by the vessels. In c) there are two possible paths for the orange vessel.

3. LITERATURE REVIEW

In this chapter, we discuss the related studies, identifying their main contributions related to the development of global and local guidance systems for USV. The focus of this chapter is to highlight commonly used guidance methods, and explain in detail COLREGS-compliant collision avoidance techniques presented in the literature.

3.1 Literature Review

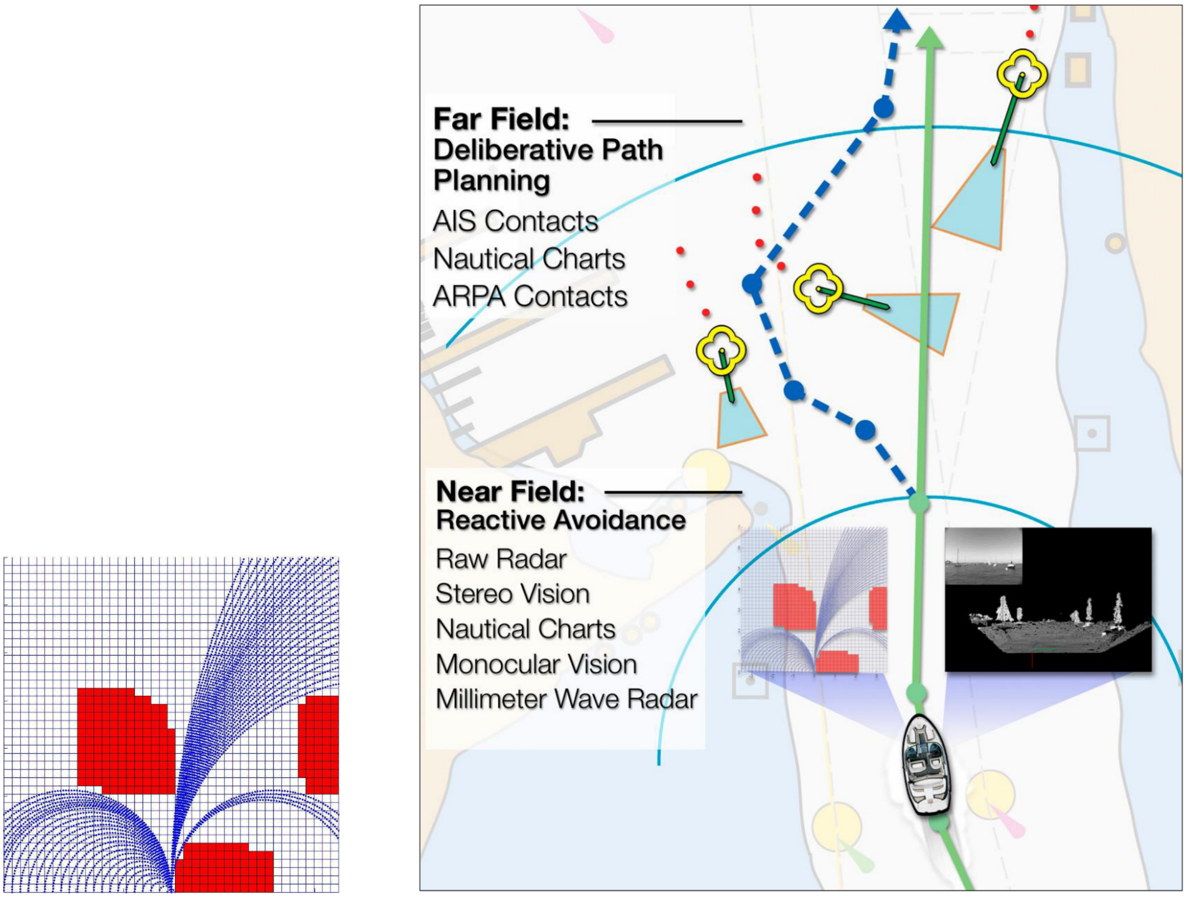
3.1.1 Larson 2006 Autonomous

Larson *et al.* [26] explore the usage of A* in combination with Velocity Obstacle (VO)[1] [15] as path planner for global guidance. A* is used to avoid stationary obstacles, and VO is used to avoid dynamic obstacles such as other vessels. The navigation is based on waypoints, and the global guidance system is responsible for the continuous modification of existing waypoints route to plan around obstacles detected with long-range sensors, such as AIS, Automatic Radar Plotting Aid (ARPA) contacts, and nautical charts. The AIS system receives position, speed, and course data broadcasts from other marine vessels with compatible systems. The ARPA system is capable of creating tracks using radar contacts and can calculate the tracked object's course, speed, and Closest Point of Approach (CPA)[2], thereby knowing whether there is a risk of collision with other ships or moving obstacles. Nautical charts can provide information about water depth, land heights, coastline, safe ways, navigational hazards, tides, currents, and human-made structures such as harbors, buildings, and bridges. Nautical charts are used as basic information for running A* and definition of waypoints. AIS and ARPA are used as basic information for running VO and definition of waypoints.

For local guidance, Larson *et al.* [26] combines the information provided by a perception system for the generation of a local world-model. The perception system for near-field obstacles detection is composed of Milimeter Wave Radar (MWR), Stereo Vision, Monocular Camera, and Laser Radar (LADAR). When an obstacle is detected, the new trajectory is defined based on the local world-model and a behavior-based approach. There are three behaviors: keep the last planned path, collision avoidance path, and free-space path. The decision is made through a voting system where the collision avoidance behavior

[1] In robotics and motion planning, a velocity obstacle, commonly abbreviated VO, is the set of all velocities of a robot that will result in a collision with another robot at some moment in time, assuming that the other robot maintains its current velocity.

[2] Closest Point of Approach refers to the positions at which two dynamically moving objects reach their closest possible distance. CPA is an important calculation for collision avoidance.



(a) Arc Path (b) Global Guidance Trajectories

Figure 3.1: Guidance System Trajectories [26]

vote is two times heavier than the others (*i.e.,*2:1:1) and generates paths in an arc format, as shown in Figure 3.1a. Figure 3.1b present a summary of Larson *et al.* work, the green path is the initial trajectory determined through A*, the blue path is the corrected trajectory for the avoidance of dynamic obstacles detected, and the red dots show the predicted trajectory of each obstacle using the VO method.

3.1.2 Agrawal 2015 COLREGS

Agrawal *et al.* [3] present an USV COLREGS-compliant path planning strategy to follow another vessel in dynamic environments. Their solution is capable of predicting the target vessel motion, and then plan the route towards the target vessel while respecting the COLREGS.

For collision avoidance, the system identifies COLREGS conditions that apply to the current situation, in a specific time, considering nearby vessels and for each vessel is calculated the CPA. If the CPA is smaller than a threshold distance but not critical, the path is re-planned using A* with restriction in the search space for compliance to the COLREGS.

When a critical distance is detected, the system re-plan using A* without restrictions in the search space.

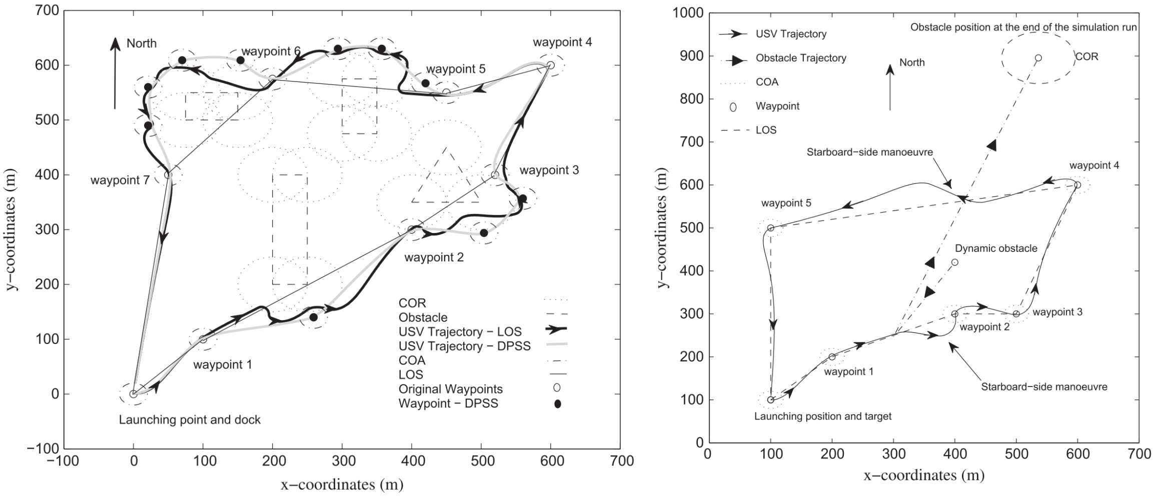
For global guidance, the path is chosen based on the predicted motion for the target vessel using Monte-Carlo sampling of dynamically feasible and collision-free paths with fuzzy weights. The prediction is continuously optimized for a particular target by learning the necessary parameters for a 3-degree-of-freedom model of the vessel and its maneuvering behavior from its path history without any prior knowledge.

For validation, they performed field tests on Panther Hollow Lake in Pittsburgh, PA, USA. They used 1.8m-long kayaks fitted with underwater-scooter motors on either side for propulsion, batteries, computers, compass, GPS, LIDAR scanners for obstacle detection, and RF receivers for manual control. The target-vessel was manually controlled at a speed of 1.7 m/s, and the presented test scenarios contain one civilian vessel that must be avoided. They successfully tested head-on and overtaking situations.

3.1.3 Naeem 2012 COLREGS

For global guidance, Naeem *et al.* [33] use an A*-based algorithm, named Direction Priority Sequential Selection (DPSS). This method is based on a goal direction vector capable of reducing the search time by up to 50%[49]. The DPSS algorithm produces waypoints around obstacles that form a smoother path with less sharp turns than A*. For local guidance, they use waypoint trajectory definition by Line of Sight (LOS) [19]. Regarding COLREGS-compliant collision avoidance, they present a manual biasing scheme consisting of always sailing towards the USV starboard side to avoid collision situations with detected objects. They argue that avoiding obstacles always heading towards the starboard side makes the system compliant to the COLREGS. However, this is not a good strategy since this action may lead to violation of other COLREGS rules such as rule 10 that impose that navigation must respect Traffic Separation Schemes (TSS). For dynamic obstacle detection, it is suggested the use of LIDAR and cameras, and for the navigation system, they suggest the use of Global Positioning System (GPS). The guidance system is evaluated through software simulation, where the control system uses a simple Proportional–Integral–Derivative (PID) for autopiloting.

For collision avoidance, Naeem *et al.* assume a Circle of Rejection (COR) around each detected obstacle (shown in Figure 3.2a). For navigation Naeem *et al.* assume a Circle of Acceptance (COA) around each waypoint. After the USV enters the COA the mission planner selects the next waypoint. Every time the distance between the USV and an obstacle is less or equal than the COR, the local guidance system becomes responsible for path plan generation. The system addresses collision avoidance for both dynamic and stationary obstacles. Figure 3.2a shows the generated trajectories using LOS and DPSS for avoid collision with static obstacles. Moreover, Figure 3.2b shows a COLREGS-compliant collision avoidance scenario, with dynamic obstacles.



(a) Static Obstacles (b) Dynamic Obstacles

Figure 3.2: Obstacles Avoidance Simulations [33]

3.1.4 Campbell 2013 Automatic

For obstacle avoidance, Campbell *et al.* [10] present a system composed of a unit to predict the next position of detected obstacles. They use the CPA method, assuming that detected obstacles will keep proceeding in their current velocities, this method finds the closest distance between encountering vessels at some time in the future. Moreover, a sampling interval is determined for updating the calculated distance. If the closest predicted distance is less than the acceptable range, the module advises a change in direction when an approaching threat is confirmed. The perception system is composed of a Microsoft HD video camera for object detection and identification and an Acuity laser range finder, so the basic information necessary for obstacle detection is extracted through the analysis of the data captured by the perception system. Moreover, the trajectory of the obstacles are predicted using Equations 3.1 and 3.2,

$$X = X_p + \int Vcos(\psi)dt \tag{3.1}$$

$$Y = Y_p + \int Vsin(\psi)dt, \tag{3.2}$$

where the pair X and Y is the predicted position, the pair X_p and X_y is the current position, and ψ is current heading angle

For global guidance, the system uses A*, and for local guidance, they use a modification of A*, named Rule-based Repairing A* (R-RA*). The focus of this work is on describing the local guidance system. The R-RA* method consists of having the path generated by A* as the base, and then perform iterative changes in the path to avoid collisions applying constraints to the search space. Path changes are performed in a local scope. The constraining is done on each iteration through the evaluation of the nodes in the search space and the addition of unwanted nodes to the A* closed list. That is, positions that violate the COLREGS are marked as belonging to the closed list. This way, A* will not consider those nodes. The world is locally represented using a grid. A core assumption of the method is that USVs can always determine other vessels' heading and velocity. The system uses this information for the calculation of the CPA. The navigation system is composed by GPS and gyrocompass. Every time the distance between the USV and the CPA is shorter than the minimum acceptable distance, the local guidance system, with R-RA* is activated. For system evaluation, they performed virtual simulations using the Virtual Sailor Simulator[3].

3.1.5 Nauss 2013 Idea

In Naus *et al.* [34] the global guidance system uses A* for path planning. Static data about the environment (*i.e.,* topography, coastline, and safe way) is extracted from electronic navigational charts, and dynamic data (*i.e.,* position of other ships) is extracted from Differential Global Positioning System (DGPS)[4], AIS, and ARPA sensors. This information is used to generate a map representation of the world and the sequential execution of the A* to determine the path to be followed by the USV. This paper does not address the problem of collision avoidance for local guidance.

3.1.6 Annamalai 2013 Comparison

Annamali *et al.* [4] focus on developing a control system but discuss the usage of a waypoint LOS scheme [19] for global guidance. The USV tries to go straight ahead from its current position until the next waypoint. For waypoint, following the control system was composed of a Model Predictive Control (MPC) autopilot. In order to decide whether a waypoint has been reached or not, the guidance system considers a COA around each waypoint. The suggested COA radius is equal to the length of the vessel. The navigation system is equipped with GPS for the determination of the current localization. For the determination of the current heading of the USV, they combine the information provided by three electronic

[3] Papini I. Virtual Sailor Software. SimSquared Ltd. Ship Simulator Shareware, 1999–2017. http://www.hangsim.com/virtual-sailor/

[4] Differential Global Positioning Systems (DGPS) are enhancements to the Global Positioning System (GPS) which provide improved location accuracy, in the range of operations of each system, from the 15-meter nominal GPS accuracy to about 10 cm in case of the best implementations.

compasses (TCM2, HMR3000, and KVH-C100). For evaluation, the system was tested in simulated scenarios. This paper does not address the problem of local guidance.

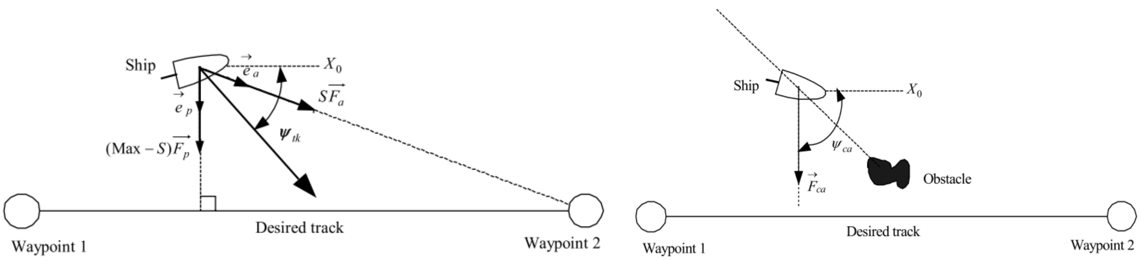
3.1.7 Lee 2004 Fuzy

Lee *et al.* [28] propose the use of Virtual Force Field (VFF)[7] for USV COLREGS-compliant collision avoidance. They present the Modified Virtual Force Field (MVFF) method, a modification of the traditional VFF algorithm based on fuzzy logic. The focus is on the determination of the desired USV heading angle Ψ_d components. The geometric terms proposed consist of track-keep Ψ_{tk}, and collision avoidance angles Ψ_{ca}. The USV route is defined by waypoints, where the path between consecutive waypoints is considered a straight line. However, movements that distance the USV from the desired path may happen, for example, when a collision avoidance movement is required. Therefore, USV motion strategy considers a combination of go towards the goal/next waypoint behavior and keep on tracking behavior. The heading angle mainly defines the motion of the USV. Lee *et al.* define the following formula for the desired heading angle:

$$\Psi_d = \Psi_{tk} + \Psi_{ca} \tag{3.3}$$

The Ψ_{tk} angle is related to a combination of the virtual forces that attract the USV, respecting a proportion, to the desired track, and towards the goal. The Ψ_{ca} angle is defined using fuzzy logic after the detection of obstacle direction, orientation, distance, relative velocity, and position relative to the USV. Each detected value is translated into a weight for fuzzy logic decision.

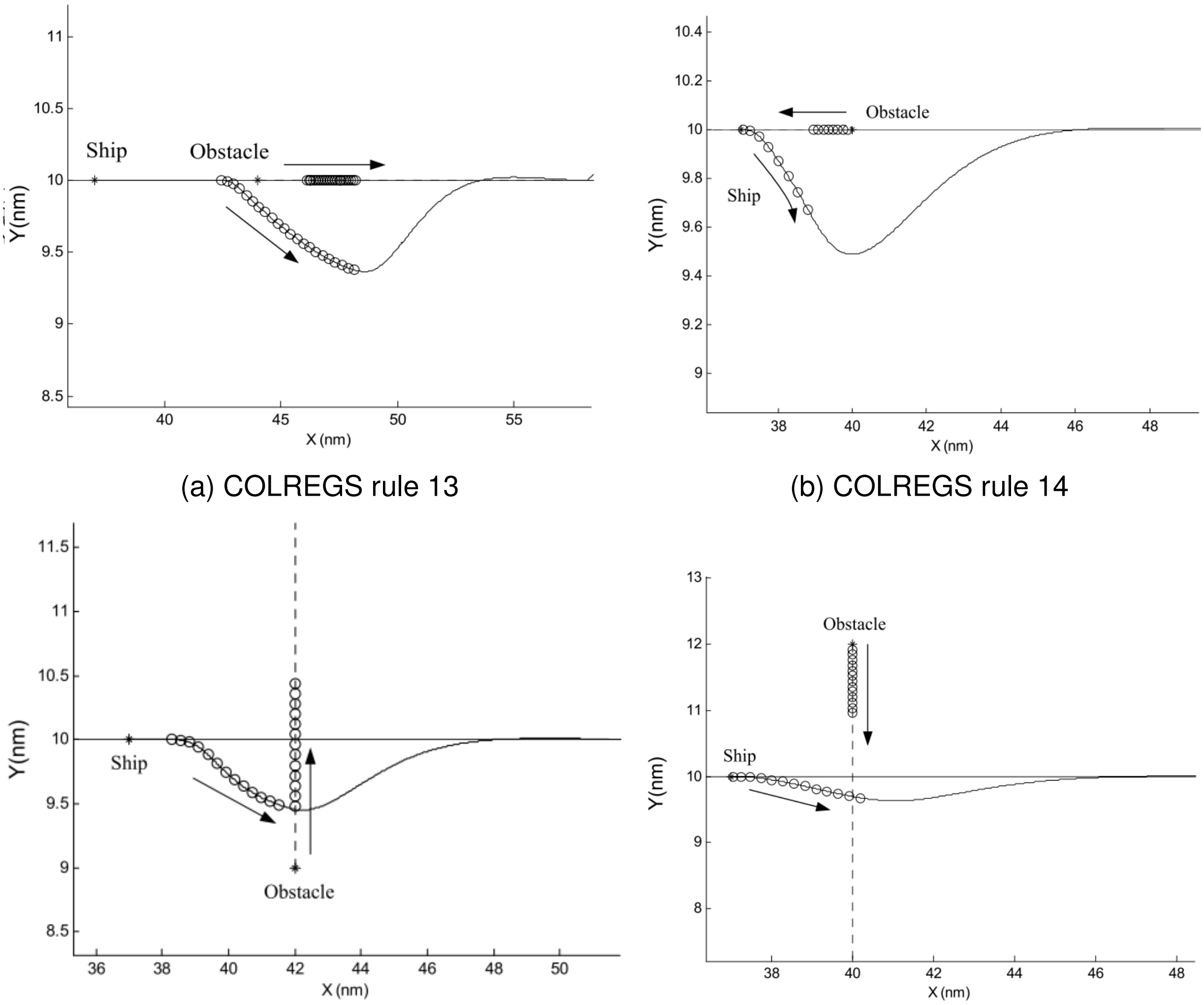
Figure 3.3a illustrates a scenario where the USV is relatively far from the desired track and must go to the Waypoint 2, $\vec{e}_a$ and $\vec{e}_p$ are the virtual forces related to the Ψ_{ca} angle. Figure 3.3b shows a similar scenario, but the USV must avoid the obstacle present. The collision avoidance angle guides the avoidance.



(a) Track-Keep Angle (b) Collision Avoidance Angle

Figure 3.3: Vectors for angles determination [28]

For simplicity, Lee *et al.* [28] assumed that the USV is capable of only turn right. Simulation tests demonstrate that the presented method is applicable for implementation of COLREGS rules 13, 14, 15, and 17 as presented in Figure 3.4.



(a) COLREGS rule 13

(b) COLREGS rule 14

(c) COLREGS rule 15

(d) COLREGS rule 17

Figure 3.4: Simulated tests [28]

3.1.8 Kuwata 2014 Safe

Kuwata *et al.* [25] present a COLREGS-compliant local guidance system based on VO capable of addressing static and dynamic obstacles. Detection of dangerous situations is done through analyses of the CPA and the geometric constraints presented in Figure 3.5, but the specific angles were not clearly defined. Environment information about static and dynamic obstacles are captured through a perception system composed of a radar, stereo cameras, and LIDAR. On real-world tests, the USV state is estimated by an onboard inertial navigation system. They discuss that VO is an intuitive way to implement COLREGS since

VO works on applying constraints to the USV possible velocity space. Therefore, apply COLREGS through VO consists of applying some more constraints to the possible velocity space. They present the constraints used to implement COLREGS rules 13 (overtaking), 14 (head-on), and 15 (crossing). For system evaluation, they executed real-world trials, and the USV was exposed to around 30 scenarios with single and multiple vessel encounters. The USV successfully acted on 24 maneuvers avoiding collision and did not respond safely to only one situation due to a vision sensor problem. They do not address global guidance system development.

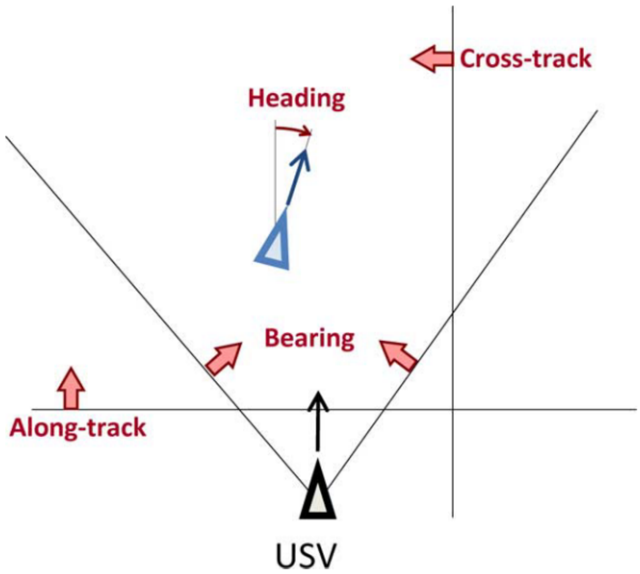


Figure 3.5: Angle Analysis for Identification of Dangerous Situations [25]

For collision avoidance on local guidance, Benjamin *et al.* [5] explore the use of a technique named Behavior-based Control and Multi-objective Action Selection. Different behaviors are defined corresponding to each possible situation, such as obstacle avoidance or keep the current mission. Each behavior defines objective functions that rates all possible actions concerning the corresponding COLREGS rule. The decision space for vehicle action is defined based on variables such as course, speed, and intended-time. Tests are performed with four real kayaks. The navigation system is capable of determining its position from GPS. Also, the USV broadcasts their positions to each other. There is no specific detection system, after receiving position information, the USV assumes that the GPS information received corresponds to another USV with which it must avoid collision.

3.1.9 Zhuang 2011 Motion

In Zhuang *et al.* [50], they declare the usage of an evolutionary path planner [41] for global guidance but do not specify which one and do not define clearly if the system is capable of dealing with static and dynamic obstacles. For local guidance, they used the VO method [15]. Dangerous situations can happen in different forms, head-on encounter, crossing, overtaking. For the identification of different situations, the system is based on geometric constraints with limits defined by angles related to the USV heading. As shown in figure 3.6, the USV heading is the 0° angle, 15° from the USV angle defines the head-on encounter sector. A crossing is defined 135° from USV heading. The system is evaluated through software simulation, and the results presented show that the local guidance system is compliant to COLREGS rules 14 and 15.

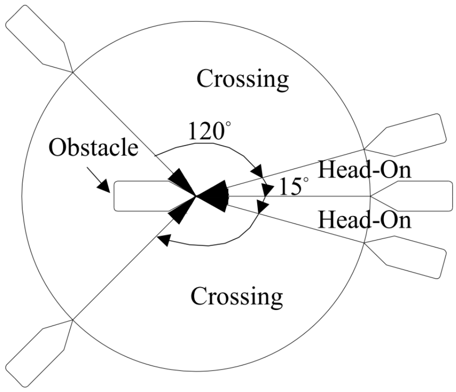


Figure 3.6: Sectors for Identification of Dangerous Situation Type [50]

3.1.10 Svec 2011

For USV trajectory definition, Svec *et al.* [45, 46] present a automated guidance plan synthesis based on Genetic Programming (GP) method [41]. An initial version of a guidance plan is generated and then improved by detecting and fixing its shortcomings. The guidance plan is improved by data mining extraction of vehicle states of failure and then new iterations using GP. Their solution is applied to the context of blocking the advancement of an intruder boat toward a valuable target. The guidance plan is represented as a composition of the navigation controller and a set of navigation plans.

The navigation controller is composed of high-level navigation commands such as *go-intruder-front*, *turn-right*, *turn-left*, and, *go-straight*, conditional variables such as *intruder-on-the-left/right/front/at-back*, *intruder-has-target-on-the-left/right*, and *usv-has-target-on-the-left/right*, standard *boolean* values and operators such as *if*, *true*, *false*, *and*, *or*, and *not*, program blocks (seq2, seq3), and system commands (*usv-sensor*, *usv-velocity*, and *usv-match-intruder-velocity*)

The navigation plan is composed of high-level navigation commands and program blocks. Leaves of the decision tree can be conditional variables or navigation commands.

Inner nodes can be conditionals variables, navigation commands, or system commands. Figure 3.7 illustrates navigation controller and navigation plan.

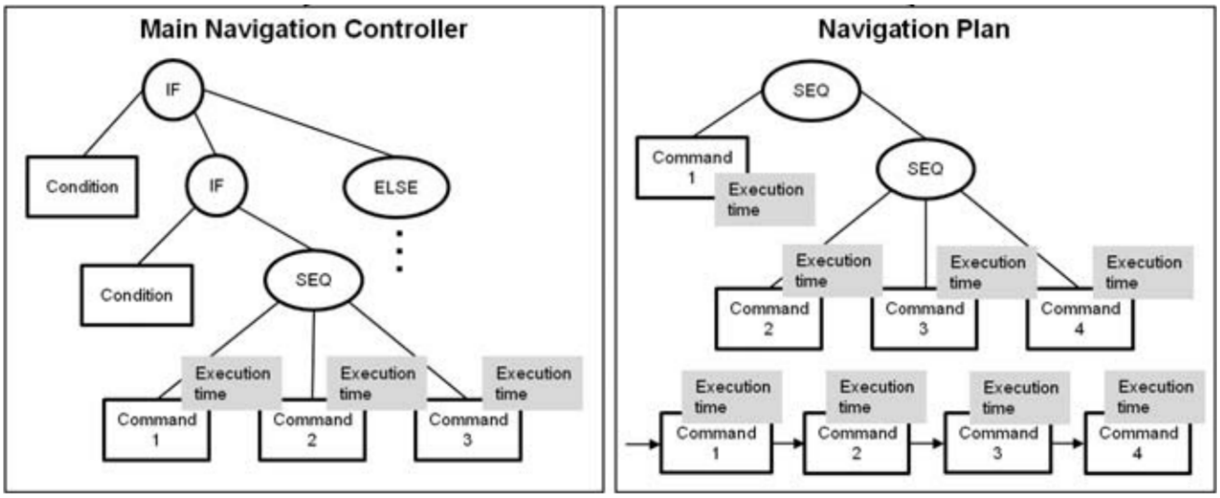


Figure 3.7: Navigation controller and Navigation plan [45, 46]

3.1.11 Soltan 2009 Trajectory

Soltan *et al.* [44] present the usage of Ordinary Differential Equations (ODEs) for the approximation of the position of obstacles through the generation of elliptical fields around the obstacles. In this study, the USV mission consists of following another target boat. The global guidance system tries to follow the target boat going towards it in a straight line. If any obstacle is detected, the local guidance system defines the trajectory around the obstacle considering the elliptical fields defined using ODEs. The evaluation of the system was done by simulation. They assumed that the USV was capable of detecting any obstacle between the USV and the target boat, in any range. Figure 3.8 shows two simulate scenarios where we can see elliptical fields for the approximation of 4 obstacles and the USV trying to follow the target boat in 2 different scenarios.

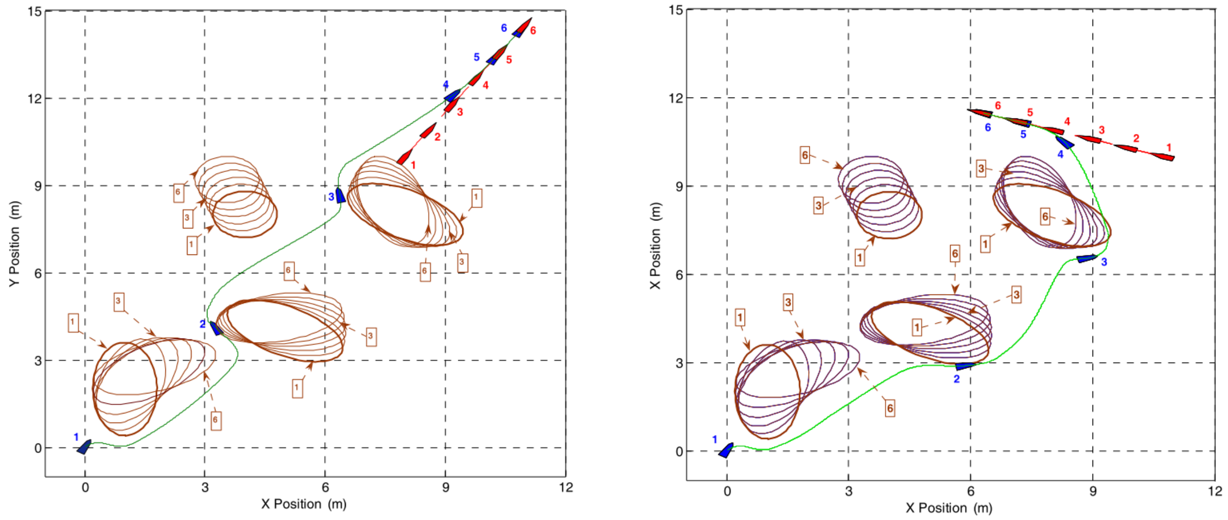


Figure 3.8: Elliptical obstacles approximations [44]

3.1.12 Abdelaal 2017 NMPC

Abdelaal *et al.* [2, 1] explore trajectory tracking controlling, based on MPC. Trying to be COLREGS-compliant, their solution prioritizes maneuvering to the starboard side when in an encounter with obstacles, but this is not a suitable approach for real-world scenarios, as discussed before. They assume to be capable of detecting and identifying velocity, course, and length of other vessels, suggesting the use of LIDAR and AIS in tests on the field. The motion prediction of encountered vessels is made using a constant velocity model [40]. Obstacles are assumed to have a circular shape, and collision avoidance is translated into an inequality constraint, integrated into the controller design. The method is validated through software simulation using MATLAB.

3.1.13 Benjamin 2004 COLREGS

Benjamin *et al.* [5] were pioneers in presenting a detailed discussion about the appliance of COLREGS to USVs. Beyond, they present an overview of the "COLREGS project," showing simulation and field test modeling. Their strategy consisted of using a behavior-based control for the integration of COLREGS-compliant collision avoidance and mission accomplishment.

For COLREGS-compliant global guidance, the output of each behavior is an objective function that rates all possible actions concerning the corresponding COLREGS. So, they use interval programming to solve the multi-objective optimization problem. The COLREGS behaviors rate possible vehicle actions in a decision space defined by the decision variables course, speed, and intended-time. Then, a not clearly defined shortest-path algorithm is run.

For validation, they performed the test in simulation and field tests. In simulation, the controlled vehicle knows its own position perfectly, as well as the position and trajectory of all other moving vessels. They present the behavior related to the head-on situation. In the in-water experiments, the vehicle knows its position from GPS, and each vehicle broadcasts its GPS position to each other. Their experiment was performed using four 10-foot kayaks equipped with on-board computers running Linux and Mission Oriented Operating Suite[16], GPS, compass, and a not clearly defined commercially available bathymetry map for the definition of free space.

3.1.14 Huang 2019 Generalized

Recently, some improvements have been made considering multi-ships dynamics. For example, Huang et al. [20] present multi-vessel collision avoidance systems using an extension of the Velocity Obstacle algorithm, namely Generalized Velocity Obstacle, for local planning. Through software simulation, it is proved that the system is capable of avoiding multiple risks encounters with other vessels considering changes on course and speed and attending COLREGS when possible. For COLREGS-compliance, the system uses hard-port turn decision. Moreover, the authors argue that the system is suitable for both autonomous and officer support.

3.2 Literature Discussion

In this section, we briefly discuss important aspects related to the development of USV guidance and collision avoidance systems presented in the literature. Based on the read studies, we can have a glance about the relevance of A*, LOS, and VO for USV guidance and the focus on addressing avoidance of overtaking, head-on, and crossing collision situations.

In general, guidance based on path planning requires addressing both stationary and dynamic obstacles. Regarding real-world-tested work, guidance requires an accurate world model and robust methods to avoid detected obstacles. World representation is done using cost maps and information is collected through nautical charts [26], electronic charts [34], ARPA contacts [26], and AIS [2]. The usage of these two last allow path planning for global guidance to consider not only stationary obstacles but moving obstacles also. Regarding the location and orientation of the USV itself, the analyzed literature presented the usage of GPS [5] and compasses [4]. Based on our literature review we identified the following methods for global and local planning:

1. Regarding global planning we identified the usage of the following methods: VO [26, 25, 50, 20], A* [33, 11, 34] e genetic programming [46, 45]. The works that presented A* and VO for global planning achieved proper behavior, being capable of planning and avoiding collision in the scenarios presented by the authors. At the same time, the usage of optimization methods (*i.e.,* genetic programming) implied limitations due to the high cost for real-time applications.

2. Regarding local planning we identified the usage of the following methods: A*[26, 10, 3], VO [26, 20], LOS [33], and Virtual Force Field [28]. Also, the review presented by Liu *et al.* [29] indicates the usage of potential fields [18, 44]. Regarding COLREGS compliance, the heuristics solutions were based on search space restriction while the solutions presented by Naeem *et al.* [33], Lee *et al.* [28], consisted in performing a hard turn to the starboard side in every detected encounter between vessels. We consider this is not a good approach since this action may lead to violation of other COLREGS rules such as rule 10 that impose that navigation must respect Traffic Separation Schemes (TSS).

Collision avoidance is performed mostly with an architecture composed of two subsystems, one deliberative and other reactive. Deliberative systems are based on well-known maps and are implemented through optimization and heuristics methods. Reactive systems run online, are based on real-time collected data from LIDAR [3], Sound Navigation and Ranging (SONAR) [12], and cameras [25] and requires more intelligent systems since they must be compliant to protocols such as the COLREGS.

Based on the literature, the identification of COLREGS situation can be made through the calculation of the bearing angle [25][5]. Some authors present the usage of CPA [10] for anticipated detection of COLREGS situation while other authors present purely reactive approaches.

Relevant work that applied and tested their techniques on real-world situations worked on using a well-known method, such as A* and VO, and applied some constraints on the search space of action to make them COLREGS-compliant [25, 10]. The COLREGS-compliant systems presented in the literature were evaluated through software simulation [44, 2, 5, 28] as well as field tests [3, 5, 25].

One of the challenges involving COLREGS is related to the fact that the rules were defined expecting a large amount of human intuition and experience to fulfill gaps and non-objective rules descriptions. For example, the vague definition of angles and distances that must be respected generate non-standard parameters definition. Studies explicitly declare

[5] We use it in our system and explain it in chapter 4

that these parameters were empirically defined after several tests [27], but these definitions not necessarily solve the problem for USV of different sizes and capabilities, generating a generalization problem.

4. COLREGS-COMPLIANT SYSTEM

Is this chapter, we present the architecture of our USV system, and detail our COLREGS-compliant local planner.

4.1 Assumptions and Limitations

The following assumptions and limitations apply for this work:

1. ROS compatibility: the whole system we developed and present in this chapter is ROS compatible, following its concepts and development paradigms. We avoid unnecessary implementation details in this chapter and focus on explaining the general operation of the system.

2. Encounters between power-driven vessels: we developed our system to run on power-driven vessels and considered the encounter between vessels of the same type. Concerning COLREGS, the encounter between different types of vessels imposes different ways to avoid collisions. Our system, in its current state, may be able to avoid collision with other types of vessels, but the avoidance strategy will always consider the other vessel as a power-driven vessel.

3. Encounter with one vessel: currently, we developed our system to perform COLREGS-compliant collision avoidance with only one vessel at a time. In a multiple vessels encounter scenario, our system may be able to perform evasive actions but we do not have any assurance regarding COLREGS-compliance in this scenario.

4. Environment perception using laser: Due the fact that perception is too complex itself, in this work we use laser detection instead of image detection with camera.

4.2 System Architecture

Our system consists of 3 main modules: guidance, navigation, and control. From a high-level perspective, the navigation system is responsible for gathering information about the environment and the vessel itself. The guidance system performs decision making using information gathered by the navigation system and then generates commands to the control system in order to achieve a goal location. The control system maps the commands received from the guidance system into actuation commands to change the our vessel's state (henceforth we refer to our vessel as Own Vessel (OV)). In Figure 4.1, we present an overview of the system, and in the following sections - 4.2.1 Navigation, 4.2.2 Control, and 4.2.3 Guidance, - we describe each module.

Beyond the GNC components of our system, Figure 4.1 shows the mission planner and gazebo modules. The mission planner module generates goals to be followed by the OV. The mission planner is an independent ROS-package[1]; we developed it as an external module. Thus, it can be replaced by any other module compliant to the standard move_base ROS goal publishing[2]. The gazebo module is the Gazebo robotic simulator [3], used by the USV_sim to interact with the objects of the simulation; through Gazebo, we can change the OV's state and gather information about an approaching vessel (henceforth referred to as Approaching Vessel (AV)) and environment state.

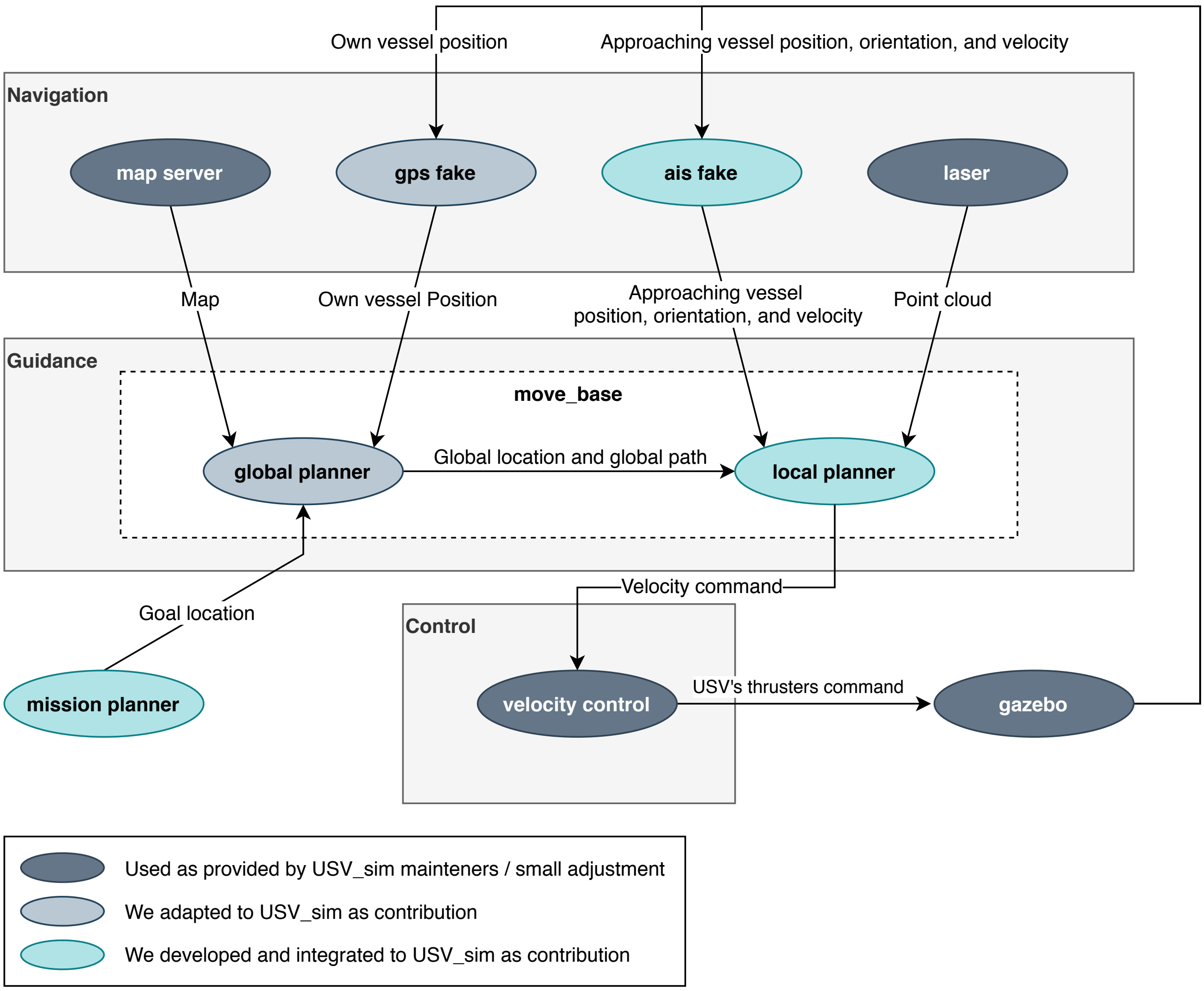


Figure 4.1: Guidance, Navigation, and Control Architecture of our System. In this Figure, we can observe the composition of the system, the messages exchanged between modules, and our contributions.

[1]ROSpackageavailableinhttps://github.com/Unmanned-Surface-Vehicle/usv_mission_planner
[2]Read more about it in http://wiki.ros.org/move_base
[3]http://gazebosim.org/

4.2.1 Navigation

The navigation system gathers data regarding environment perception and state estimation. The navigation system consists of the following sub-modules:

- **map server**: the map server module is responsible for making the global map available for the guidance system. The map is generated before the start of the OV mission. Both the map and an adapted version of the map server module were made available by the USV_sim maintainers.
- **gps fake**: the gps fake module is a simplification of a real GPS; its information is extracted directly from the simulator; that is, it does not perform a simulated query to satellites. The gps fake module performs a simple query to the gazebo module in order to acquire a 3-tuple - (x, y, z) - for determination of the position of the OV in the global reference frame. This module can be replaced by another localization method, such as Adaptative Monte Carlo Localization (AMCL)[4] or a real GPS device. We integrated this module into the system as a contribution of this work.[5]
- **ais fake**: the ais fake module is a simplification of the real AIS system[6] used on vessels sailing on the high seas. The ais fake module provides the position and velocity of an approaching vessel. The ais fake generates its information from a direct query to the simulator data. We developed and integrated this module into the system as a contribution of this work.
- **laser**: this module provides the location of bodies of mass that reflect the laser light beam through an ordered 3-tuple - (x, y, z) - position referring to the global frame. In our system, the laser module performs detection of static and dynamic obstacles, and every detection generates an update on the local cost map (automatically done by the ROS move_base). This module is available as a standard module in USV_sim. We configured this module to be compatible with the following requirements: laser beam range up to 25m and 360° detection capability; compatible for both specification with real lasers, for example, the Slamtec RPLIDAR A3 laser [42].

[4] See more in: http://wiki.ros.org/amcl

[5] Implementation based on http://wiki.ros.org/fake_localization

[6] A real AIS module must respect NMEA legislation, refer to NMEA for specification: https://www.nmea.org/Assets/nmea%20collision%20avoidance%20through%20ais.pdf

4.2.2 Control

The control system is composed of the velocity control module. The velocity control module maps velocity commands received from the local planner into actuation commands. It interacts directly with the simulator and sends actuation commands to modify the OV's state. This module is available as a standard module in USV_sim, and we used it without any modifications.

4.2.3 Guidance

As shown in Figure 4.1, the guidance module is mainly composed of global and local planners. We implemented both planners inside the ROS move_base environment[7]. For global planning, we use move_base standard A* search[8], and for local planning, we developed our COLREGS-compliant A*. The global planner determines the global route, and the local planner is responsible for generating velocity commands to move the OV, regardless of whether there are obstacles nearby or not - the moves_base ROS structure imposes this style.

Apart from the planners, the guidance module has cost maps. Cost maps are data structures composed of cells that store information about each position of the environment. There is a cost map for each planner, and each cell of a cost map has a cost value (range from 0 to 255, where a cost greater than 250 represents an obstacle). Thus, the planners can search for paths. In Figure 4.2 we can observe a practical exemplification of the difference between global and local cost maps. Around the robot location, we can see a squared grid with a gradient from blue to red, indicating proximity to obstacles, this squared grid is a local cost map. All information outside the local cost map composes the global cost map. In our system, the obstacle detection using laser to modify the local cost map is done by the ROS move_base.

In our system, the local cost map is a 100x100 grid (size based in Agrawal *et al.* [3]), representing a 20mx20m area with 1:0.2 resolution, resulting in a search space of 10000 cells. The local cost map has our COLREGS-compliant vessel at its center with some variation of at most two grid cells due to approximation errors when converting global position to local position.

[7]The ROS move_base environment is a ROS framework that allows the standardized creation of robotics applications regarding robots motion.

[8]http://wiki.ros.org/global_planner?distro=kinetic

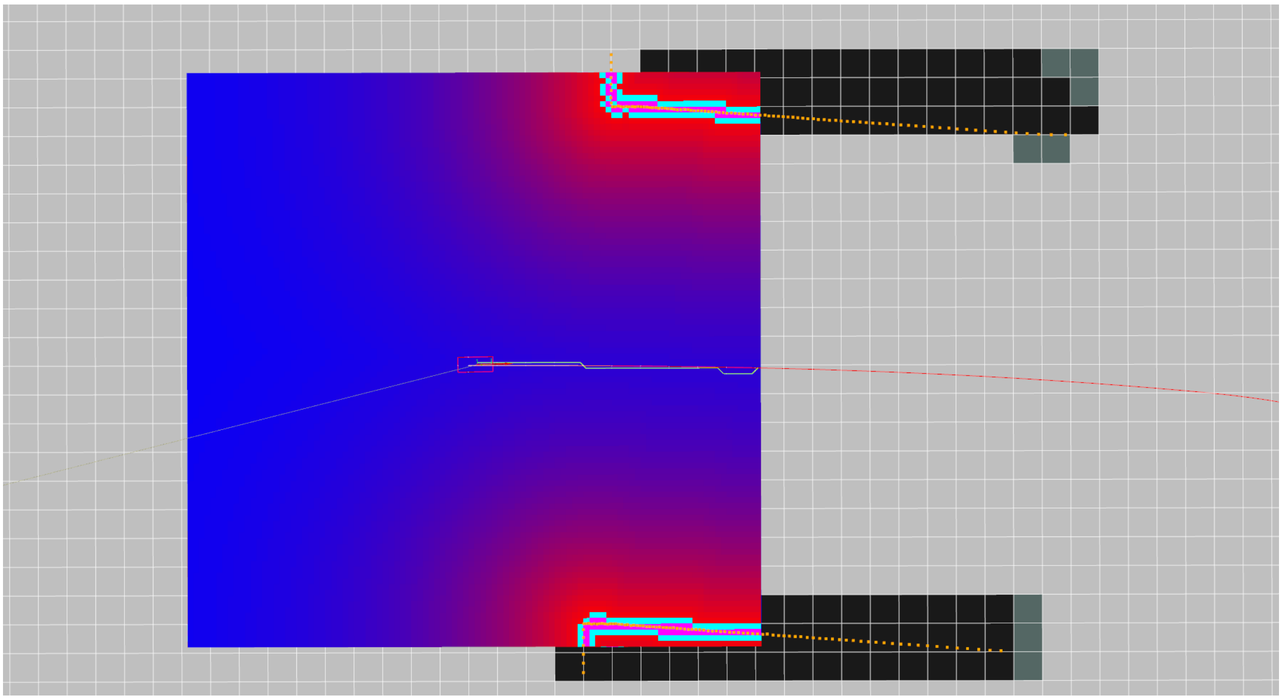


Figure 4.2: Simulation regarding the comparison between global and local scopes. In the left half of the Figure, surrounded by a red box, we have our COLREGS-compliant vessel capable of detecting obstacles in its proximity using a laser. Every orange dot represents a laser contact in an obstacle's surface. In black, we see globally known obstacles. Around the robot position, we can see a squared grid with a gradient from blue to red, indicating proximity to obstacles. This squared grid is a local cost map. All information outside the local cost map composes the global cost map. Moreover, we show global and local paths, respectively, in red and green.

4.2.3.1 Local Planner

The local planner module is responsible for the local path planning and the reactive behavior of our system. When encountering static or dynamic obstacles within the local cost map area, the local planner reacts generating collision-free local paths, while maintaining alignment with the global route generated by the global planner. For pathfinding, our local planner performs A* search in the local cost map. Performance measurement is presented in the next chapter.

Our local planner performs COLREGS-compliant path planning using a modified version of Artificial Terrain Cost (ATC) presented by Agrawal *et al.* [3]. With ATC, we restrict the search space, building virtual obstacles in no COLREGS-compliant locations. The path planner creates virtual obstacles from the AV location until the local cost map border. Virtual obstacles increase the cost of cells to a value above the A* acceptance. Thus, when searching for routes to reach the goal location, A* is unable to choose regions that violate COLREGS. Once Agrawal *et al.* [3] used A* as path finding method, and our approach consists on using Agrawal *et al.* [3] solution, we implemented A* as path finding in our local planner.

In Figure 4.3, we present the creation of virtual obstacles for each of the four encounters discussed in COLREGS. For head-on and crossing from the right encounters, the local path planner creates the virtual obstacle in the port side, forcing the planner to determine a route through the OV's starboard side, featuring COLREGS-compliant behavior. For overtaking, COLREGS specifies that the vessel performing overtaking should not generate another crossing encounter during its operation. Thus the local planner creates a virtual obstacle in the direction in front of the AV. For crossing from left, no virtual obstacle is created once COLREGS specifies that in this situation, the vessel coming from left is responsible by the avoidance, and the other vessel must stand-on its current course.

Virtual obstacles occupy an area $w \times m$ where w is the width of our vessel in the local frame (2 local cost map cells), and m is the distance between the target vessel and the corresponding border. In the worst-case, m is 99 since the local cost map side is 100, and at least one cell needs to be occupied by the target vessel to trigger the ATC generation.

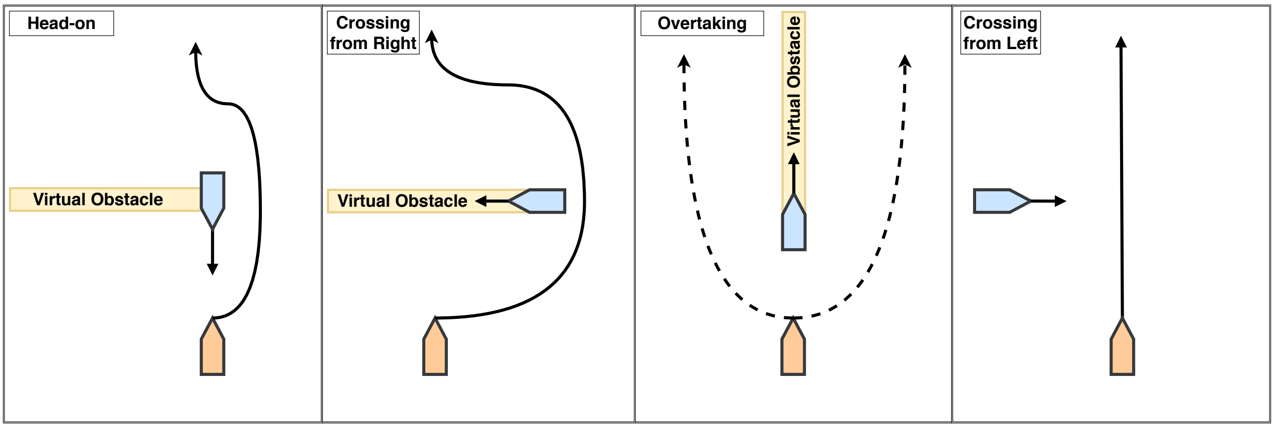


Figure 4.3: Virtual obstacles creation for each encounter discussed in the COLREGS. In blue, we present the approaching vessel and in orange we present our COLREGS-compliant vessel.

The local planner generates, at every computation cycle, a velocity command for the vessel, which is composed of linear and angular velocities. The velocity command is sent to the control system and determines the local behavior of the OV. The local planner generates the velocity command based on the path found after executing the ATC A* search method. The local planner apply a transfer function based on proportional gain that generates the velocity command considering the next desired pose[9] (x, y, z, θ) and the current pose of the OV[10]. In Equation 4.1, we show linear and angular velocities compositions. We empirically defined values for K_{p_l}, and K_{p_a}. Linear velocity is proportional to the distance between the goal and OV's current position. While angular velocity is proportional to the difference between the OV's current angle and the steering angle[11] [38] regarding the goal and OV's current position.

[9]A pose is composed of position and orientation.
[10]We developed the transfer function based on http://wiki.ros.org/turtlesim/Tutorials/Go%20to%20Goal
[11]http://street.umn.edu/VehControl/javahelp/HTML/Definition_of_Vehicle_Heading_and_Steeing_Angle.htm

$$Vl = K_{p_l} * G_l$$
$$Va = K_{p_a} * G_a \tag{4.1}$$

where Vl means linear velocity, Va means angular velocity, $K_{p_l} = 0.075, K_{p_a} = 0.75$,

$$G_l = \sqrt{(x_g - x_c)^2 - (y_g - y_c)^2}$$

, and

$$G_a = atan2\ [43](y_g - y_c, x_g - x_c) - \theta_c$$

where g denotes goal position, and c denotes OV's current pose.

In Figure 4.5 we summarize our planner's operating sequence. Based on the decisions we enumerated the following list:

1. The proposed local planner waits to receive a mission. As discussed earlier, missions are location goals generated by the mission planner module.

2. After receiving a goal, the planner evaluates information received from the ais fake module. If there is a vessel nearby, the local planner identifies the type of COLREGS encounter and then creates a virtual obstacle by changing the local cost map.

   As done by Benjamin *et al.* [6], Loe [31], and Xu *et al.* [48], the relative bearing[12] [17] β between the OV and AV is used to identify the type of COLREGS encounter (*i.e.,* head-on, crossing, or overtake) when two vessels approach one another. Figure 4.4 illustrates this list of angles defined by Benjamin *et al.* [6], Loe [31], and Xu *et al.* [48]. These angles are not defined by the COLREGS and they change according to the authors. We selected these angles presented below since they are the most used in the literature.

   - Head-on: $\beta \in$ [-15.0°, 15.0°);
   - Crossing from the right: $\beta \in$ [15.0°, 112.5°);
   - Crossing from the left: $\beta \in$ [-112.5°, -15.0°);
   - Overtaking: $\beta \in$ [112.5°, 180.0°) $\cup$ [-180.0°, -112.5°).

[12] https://revisionmaths.com/gcse-maths-revision/trigonometry/bearings

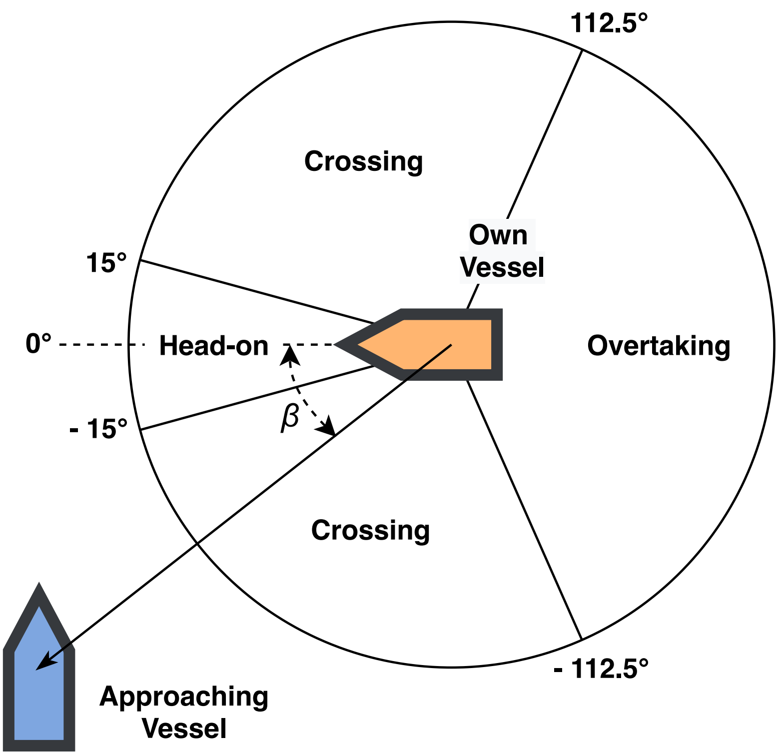


Figure 4.4: Encounter type identification. In this Figure we see our vessel encountering another vessel by its port side, featuring a crossing from left encounter.

The relative bearing is calculate as

$$\beta = atan2(y_{ov} - y_{ev}, x_{ov} - x_{ev}) - \theta_{ev} \tag{4.2}$$

where θ_{ev} is the heading angle of the AV.

3. The local planner validates if the local goal can be used as a goal for the A* method. A goal is considered not valid when an obstacle occupies its same location. In that case, the local planner searches for another possible local goal within the local cost map, through a reverse search into the global path. If the goal is valid, we run A*.

   The local goal is defined as being the farthest location in the path defined by the global planner within the local cost map (see Figure 4.2);

4. After A*, a path may have been generated or not. If there is a path, the planner generates a velocity command;

5. If the last execution of ATC A* had a valid goal, the local planner uses the last velocity command. If there is no last valid goal, the local planner generates a stop command.

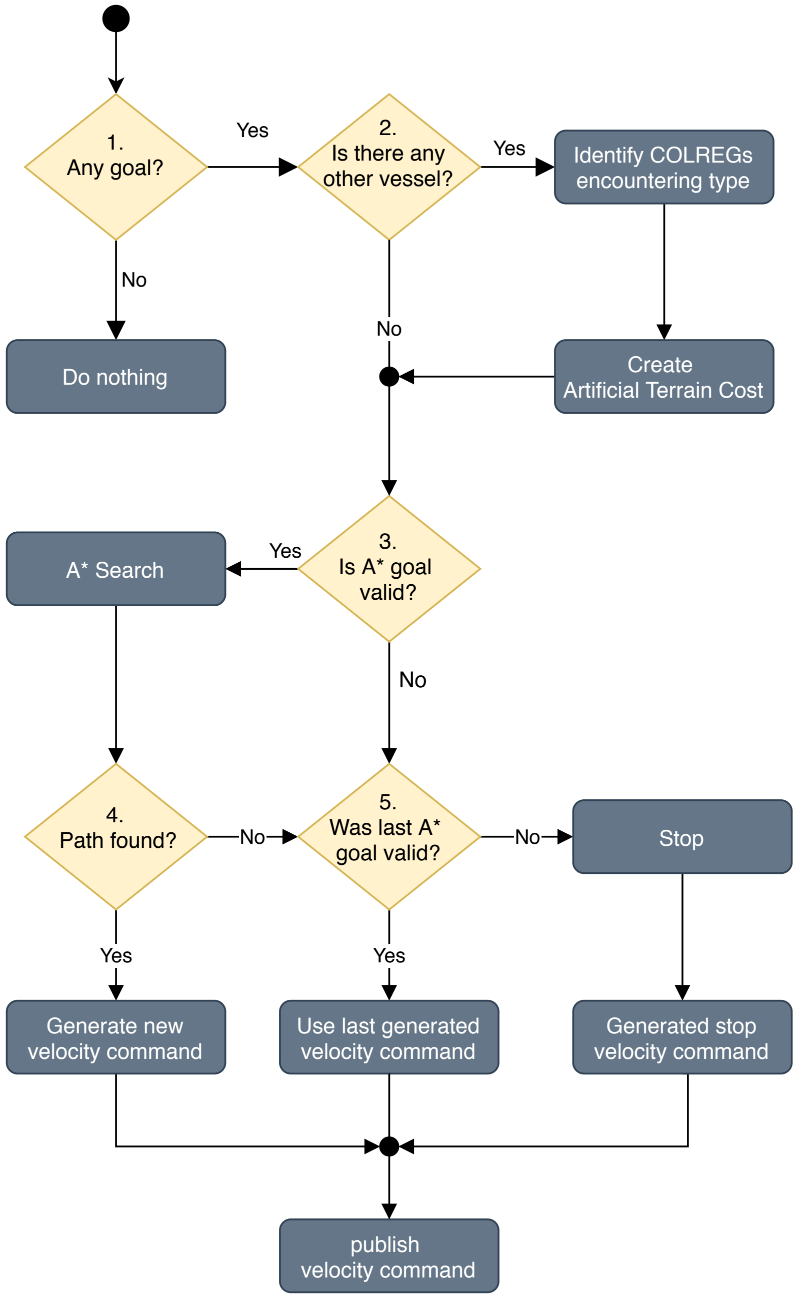


Figure 4.5: Here we present a simplification of the sequential execution of our COLREGS-compliant path planning method.

Regarding holonomy, our path planner consider the systems is non-holonomic. After a path is defined the system rotate till the difference between the heading angle and the goal position is lower than 60 degrees, then the system goes ahead in direction to the next goal. This way any path generated by our path planner can be followed by the non-holonomic vessel we used.

5. SIMULATION AND RESULTS

In this chapter, we present simulation scenarios for qualitative and quantitative evaluation of our system, as well as the experimental results. The simulations cover four different encounter scenarios between two vessels (head-on, crossing from the right, crossing from the left, and overtaking). For each encounter, we apply three variations, the first one uses our solution, the second does not use our solution, and the third uses our solution and is exposed to wind. For qualitative evaluation, we analyzed the final trajectories, avoidance success, and the behavior of our system when exposed to different environmental conditions (we only vary wind speed). For quantitative evaluation, we analyzed the time for computation of our path planning method and the minimum distance kept between our COLREGS-compliant vessel and the encountering vessel during the simulation. We observed that the variability of the system and scenarios is low, this way, we consider that a sample of our tests for each case, is representative.

5.1 Simulations Characterization

We run our simulations on USV_sim[1] [35] simulator using the platform described in Table 5.1. In our simulations, we use a differential boat - shown in Figure 5.1 - with two thrusters, which enables it to rotate over its axis. This boat is modeled according to specifications of the Lutra Prop boat, acquired from Platypus [30]. Beyond the specifications shown in Table 5.2, the Lutra boat we use in our simulation has a laser rangefinder for environment scanning in its bow. We set the rangefinder to be capable of detecting objects within 25 meters in a range of 360°.

[1] https://github.com/disaster-robotics-proalertas/usv_sim_lsa

Table 5.1: Simulation Platform Specification

Component	Specification
Computer	Desktop Dell XPS 8700
Processor	Intel® Core™ i7-4770 CPU @ 3.40GHz × 8
Memory	Teikon PC3-12800u DDR3 1600 MHz 2GB x 2 Teikon PC3-12800u DDR3 1600 MHz 4GB x 2
Operating System	Ubuntu 16.04.6 LTS
ROS Version	ROS Kinetic

Figure 5.1: Simulated version of Lutra Prop boat

Parameter	Value
Length	106 cm
Width	48 cm
Height	15 cm
Weight	9.7 Kg
Maximum speed	1.41 m/s

Table 5.2: Lutra Prop parameters

We assembled the scenarios in a simulated version of the Dilúvio stream. The Dilúvio stream is a potential location for real-world trials of our system since it is near our laboratory, so we evaluate the behavior of our system on a simulated version of it. The maintainers of the USV_sim created this scenario. It was built by combining geographic location and relief information with geometry information and 3D buildings. This scenario simulates a real area of 1340mx555m. In Figure 5.2 we show with a red box the specific location[2] we simulated in our experiments and its simulated version.

As done by several authors (*e.g.,* Larson *et al.* [26], Naeem *et al.* [33], Campbell *et al.* [10], Naus [34]), for evaluation of the COLREGS-compliance of our system, we assembled 4 main encounter scenarios between two vessels (head-on, crossing from the right, crossing from the left, and overtaking). In our scenarios one of the vessels has our

[2]Google maps location: (-30.047258°, -51.232660°), Av. Edvaldo Pereira Paiva, 1970 - Praia de Belas - Porto Alegre - RS - Brazil

(a) Real World Location (b) Simulated version of Real World Location

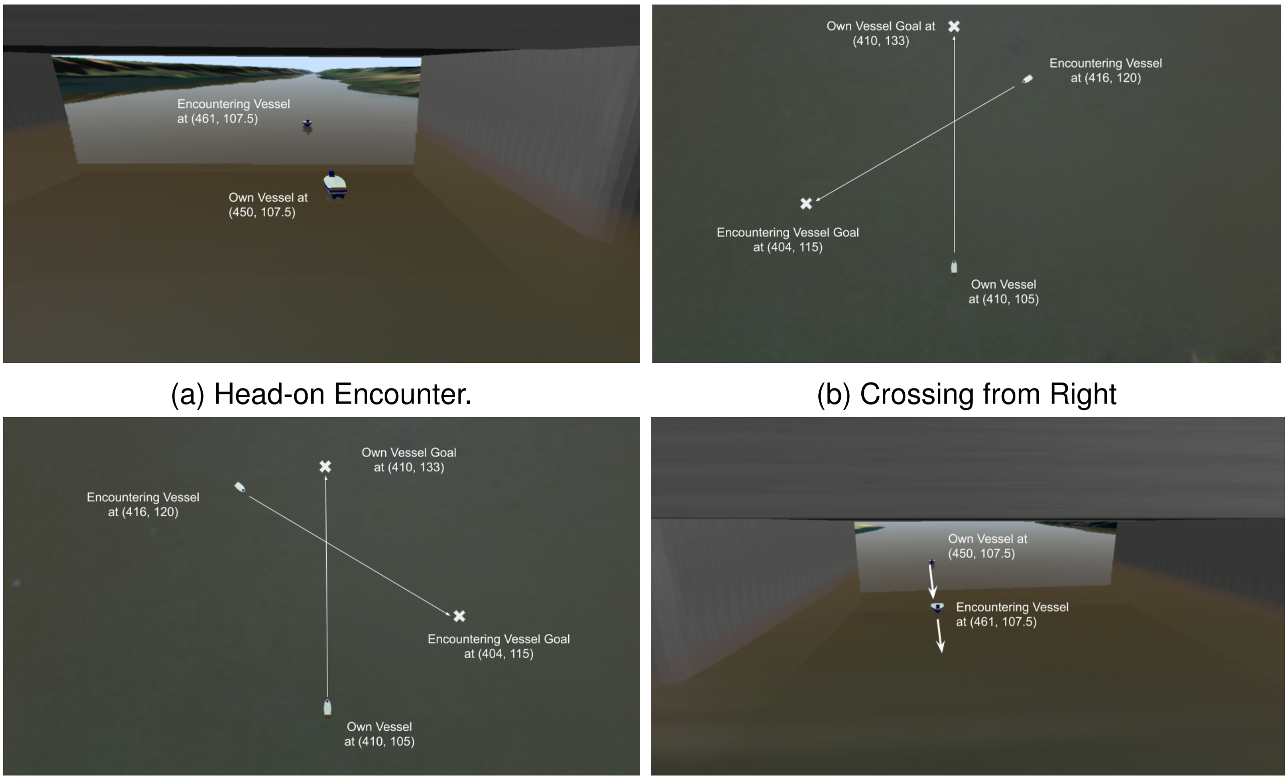
Figure 5.2: Real world region and its simulated version

COLREGS-compliant system (from now on referred as Own Vessel[3] (OV)) and the other vessel (from now on referred as Approaching Vessel (AV)) don't have any COLREGS compliance knowledge. The starting configuration of each vessel in each scenario is described in Table 5.3 and shown in Figure 5.3. Both head-on and overtaking scenarios, were assembled right below the bridge shown in Figure 5.2, while the crossing scenarios were assembled far the bridge, due to the need of more space between the start positions.

Table 5.3: Own Vessel and Approaching Vessel Encounter Scenarios Configuration

	Own Vessel		Approaching Vessel	
Encounter Type	**Initial Pose (m, m, º)**	**Target Position**	**Initial Pose (m, m, º)**	**Target Position**
Head-on	(450, 107.5, 0)	(480, 107.5)	(461, 107.5, 180)	(350, 107)
Crossing Right	(410, 105, 90)	(410, 133)	(416, 120, 215)	(404, 115)
Crossing Left	(410, 105, 90)	(410, 133)	(404, 105, 315)	(416, 115)
Overtaking	(450, 107.5, 0)	(600, 107.5)	(461, 107.5, 0)	(600, 107)

[3]Same nomenclature used by Naeem *et al.* [32], He *et al.* [17], and other authors.



(a) Head-on Encounter. (b) Crossing from Right

(c) Crossing from Left (d) Overtaking

Figure 5.3: Encounter scenarios for evaluation. Scenarios adapted from Huang *et al.* [20].

5.2 Experiments Results

We qualitatively evaluate the behavior of our method in two different configurations. In the first configuration, we compare the behavior of the system with and without ATC. In the second configuration, we compare our system's behavior with ATC with and without the influence of wind. We executed both scenarios for four types of possible encounters between the two vessels. For the quantitative evaluation of each scenario, we measured the computation time of every execution of our path planner, and minimal distance maintained between the vessels, as well as whether the collision avoidance was successful or not. In Table 5.4, we summarize the collected results.

5.3 Head-on

In Figure 5.4, we present, comparatively, the final trajectory of two vessels in two executions of the same head-on scenario. For the OV, the performed trajectory is guided by our local guidance system, while the AV is following a path towards a given goal ahead. In the "ATC case" execution, our system is fully functional, in the "no ATC" execution we removed the ability of the planning system to generate virtual obstacles, which is the core of the ATC method and partially responsible for the COLREGS-Compliant path planning. In both runs, in the time interval from t0 until just before t1, OV goes northeast, due to a static obstacle located from (450, 104) to (460, 104).

Until t1, OV tends to distance itself from the static obstacle. At t1 for both executions of the scenario, AV becomes noticeable at the OV local cost map. From t1 onwards, OV reacts differently to each scenario. We observe that even with the existence of a static obstacle to the south in its proximity, OV in the ATC case decides to avoid the encounter with the other vessel moving to its starboard side, featuring a COLREGS-Compliant behavior. OV in "no ATC" case, is influenced only by the existence of an obstacle in the south and at the encounter with AV, performs a not COLREGS-compliant trajectory.

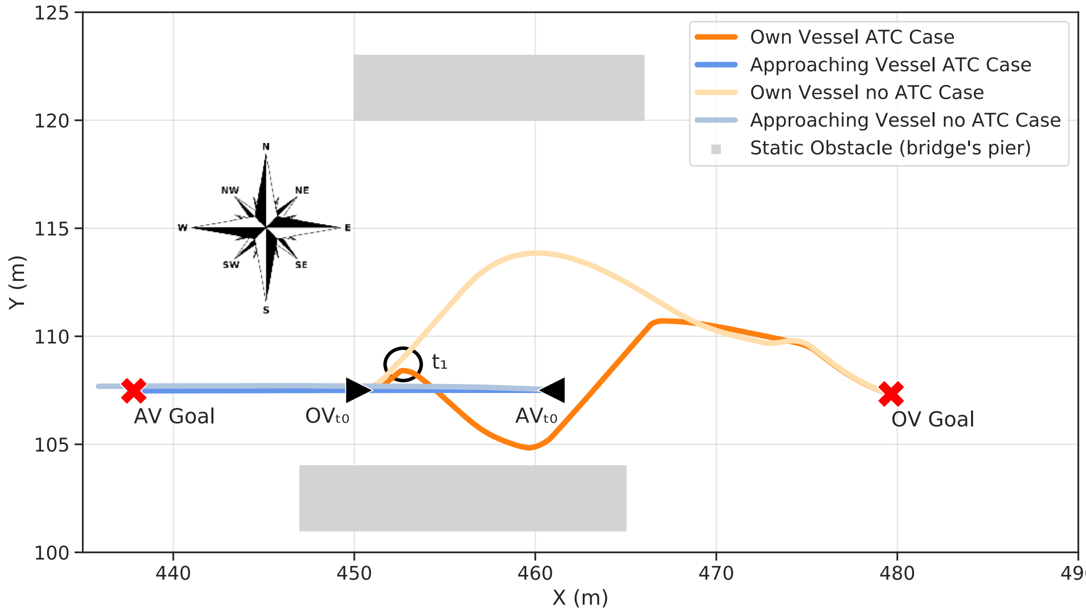


Figure 5.4: Comparison of vessel trajectories in ATC case and no ATC in a head-on encounter scenario. Start position for OV and AV are marked with "►" and "◄", while their goals are marked with "x". In gray, we present static obstacles. In blue, we show the trajectory done by the AV. In orange, we show the trajectory of the OV with our path planner. In yellow, we show the trajectory of the OV without our path planner. We can observe that our ATC A* method implied COLREGS compliance, since on the detection of another vessel (at time t_1) it avoided collision going to its starboard side.

Figure 5.5 shows the computational time measured in seconds over time. Our COLREGS-Compliant ATC A* method had a peak cost of 0.364s for path generation for this head-on scenario. As we can see, both ATC and no ATC cases have similar computational cost curves. This happens because most of the computational cost of our path planning method is related to our A* implementation. The ATC method alone has low computational consumption. The worst-case scenario consists of filling a grid area of dimension $w \times m$, where w is equals to OV's width (2 local cost map grid cells), and m is the worst case 99. Over time the computational time reduces due to the proximity to the goal position.

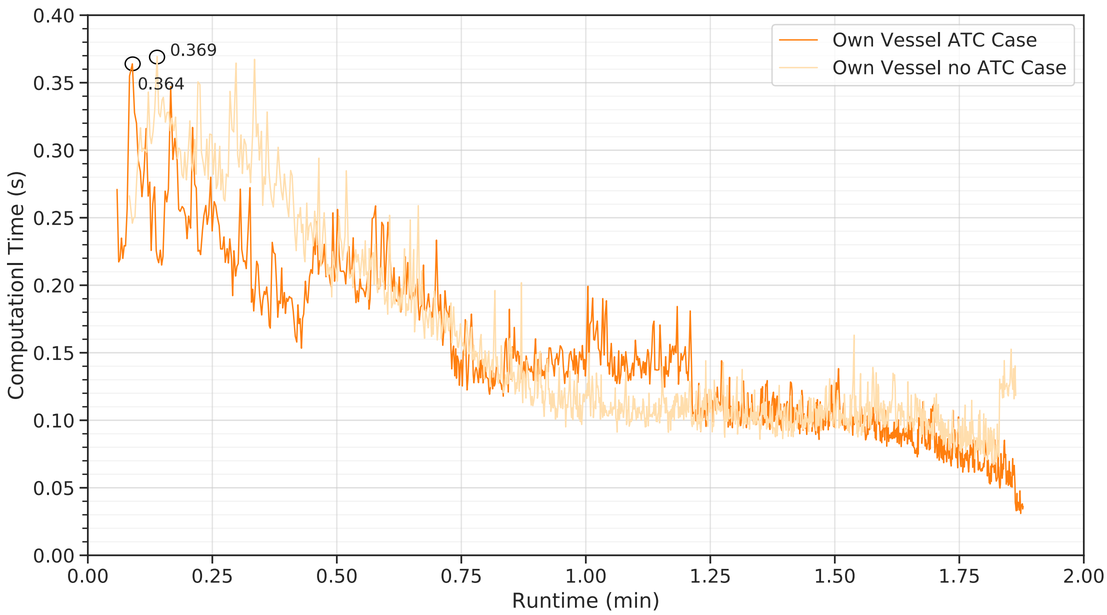


Figure 5.5: Computational time comparison between ATC case versus no ATC in a head-on encounter over time. The system achieves a peak cost of around 0.364s using ATC and around 0.369 without ATC.

In Figure 5.6 we compare final trajectories for the same head-on scenario described in 5.3, now for two executions of the simulation with different wind influence, in both we use ATC A*. "No wind case" shows the behavior of the OV being influenced by no wind. "Wind case" shows the trajectory of our OV being influenced by the wind with northeast direction and intensity of 2.0 m/s, indicated by an arrow. We can see the change in the trajectory of OV and that it still maintains COLREGS-compliance. AV response to the wind is related to the standard control system used; it seems not to be capable of acting against the 2 m/s wind intensity. In this work, we do not evaluate the limitations of the AV's control system. We empirically determined a 2.0 m/s for evaluation of our system. In all simulations using 2.0 m/s for wind intensity, our system was capable of reacting and avoid the collision. With wind intensity greater than 2.0 m/s, our system was not capable of avoiding collision, being strongly influenced by wind. This behavior is related to our internal controller (inside local planner and presented in 4.2.2).

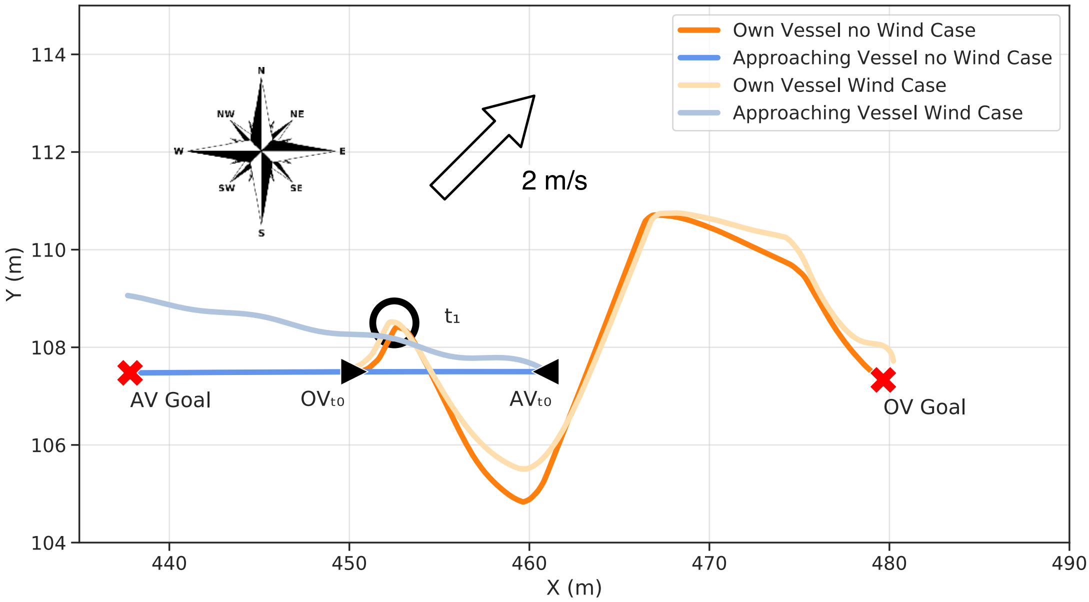


Figure 5.6: Comparison between the trajectories of the vessels in Wind case and no Wind case in a head-on encounter scenario, both cases use ATC A*. Start position for OV and AV are marked with "►" and "◄", while their goals are marked with "x". For the Wind case, the direction of the wind is northeast, represented by an arrow. We can observe that our ATC A* kept implying COLREGS compliance even under the influence of wind in this scenario.

Figure 5.7 shows the computational time measured in seconds over time for wind and no wind cases. We observe that the computational time remains stable and similar to the previously presented cases. The computational time does not seem to be related to the imposed wind influence for this simulation.

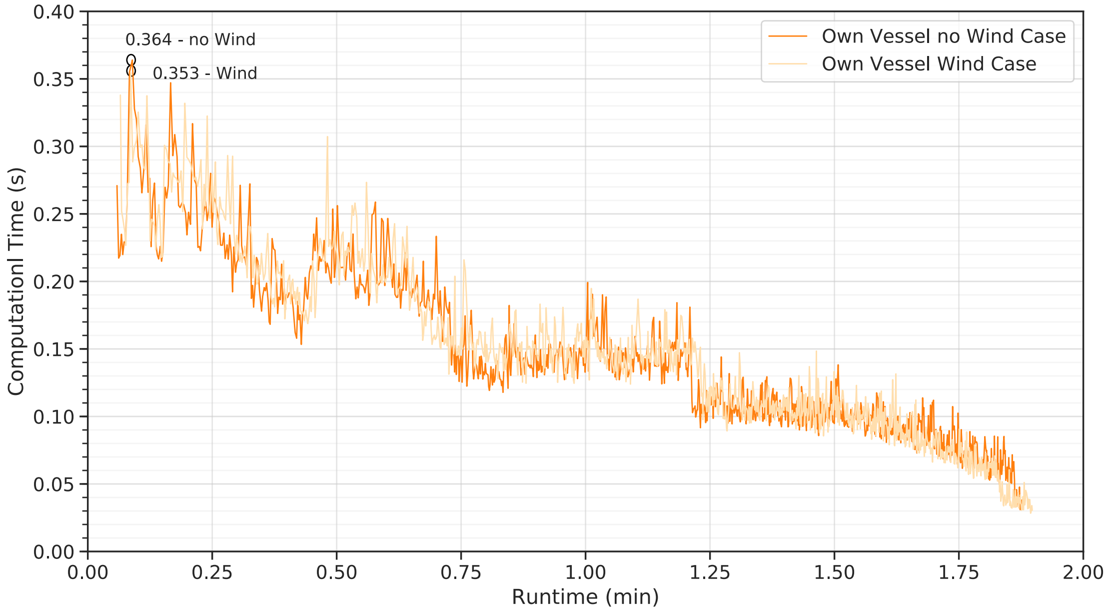


Figure 5.7: Computational time comparison between Wind case versus no Wind in a head-on encounter. The system achieves a peak cost of around 0.364s using ATC without wind and around 0.353s with wind.

5.4 Crossing from the Right

In Figure 5.8, we present the behavior of our system when the OV encounter another vessel coming from the right side. In Figure 5.8, we show the comparison between trajectories with and without ATC. As we can see, our method implies COLREGS compliance when avoiding the collision. Regarding the trajectories, at time t_1 for each of the scenarios, the OV has different behaviors. From the start until t_1, for both scenarios, OV goes to the northwest direction. From t_1, OV detects AV in its local cost map. In the scenario with our local planner, OV decides to avoid collision going to its starboard side, featuring a COLREGS-compliant behavior. While for the scenario where OV do not have our local planner, it decides to keep going northwest.

Figure 5.9 shows the computational time measured in seconds over time. Our COLREGS-Compliant ATC A* method had a peak cost of 0.395s for path generation for this crossing from the right scenario. As we can see, both ATC and no ATC cases have similar computational cost curves.

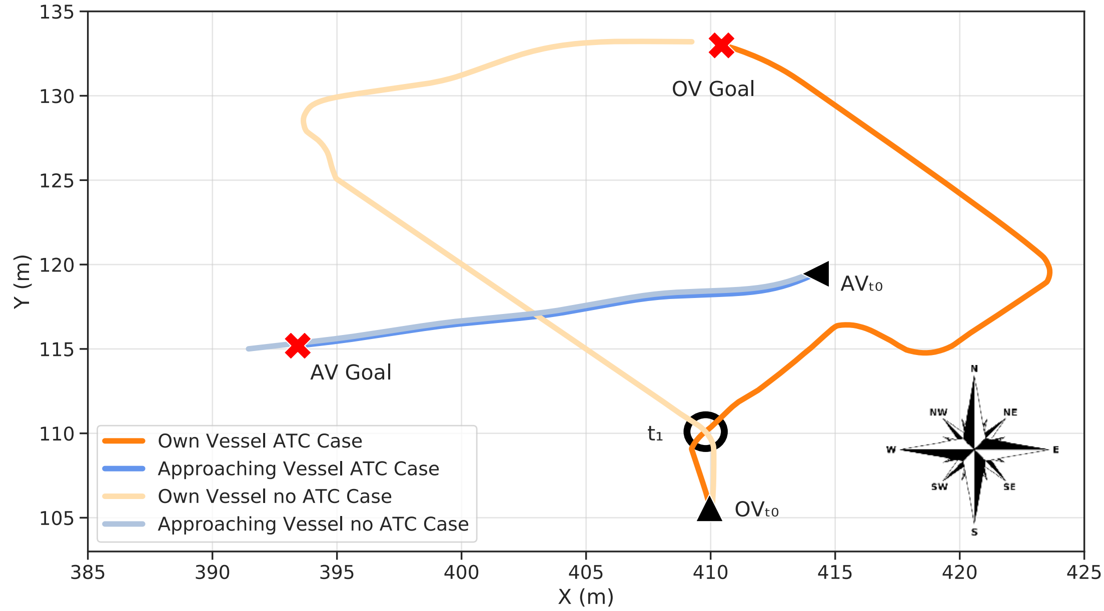


Figure 5.8: Comparison of vessel trajectories in ATC case and no ATC in a crossing from the right encounter scenario. Start position for OV and AV are marked with "▲" and "◄", while their goals are marked with "x". In blue, we show the trajectory done by the AV. In orange, we show the trajectory of the OV with our path planner. In yellow, we show the trajectory of the OV without our path planner. We can observe that our ATC A* method implied COLREGS compliance, since on the detection of another vessel (at time t_1) it avoided collision going to its starboard side.

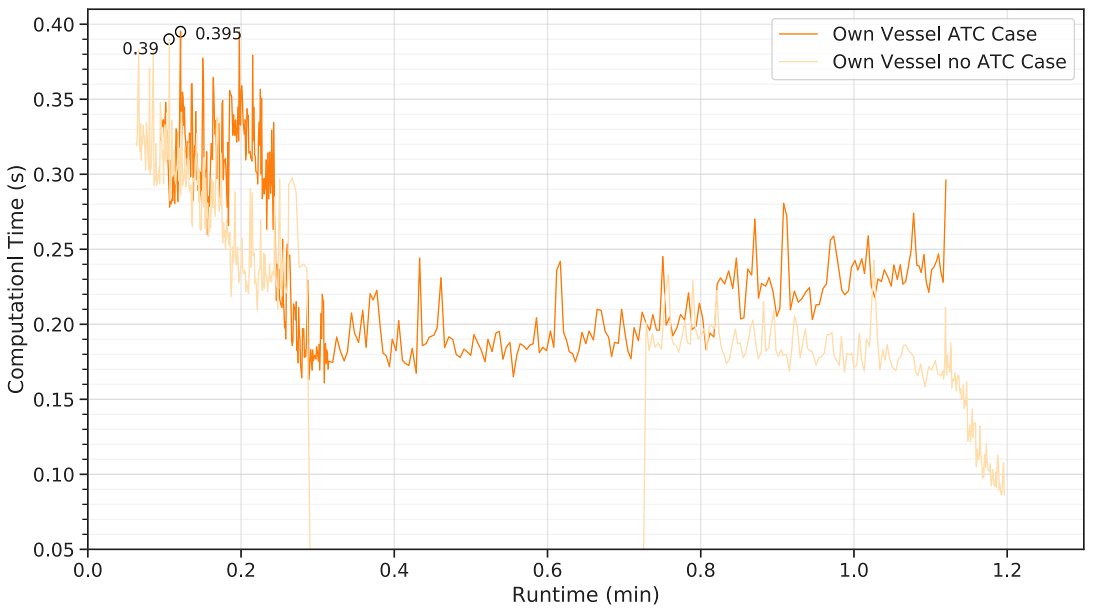


Figure 5.9: Computational time comparison between ATC case versus no ATC in a crossing from the right encounter over time. The system achieves a peak cost of around 0.395s using ATC and around 0.390 without ATC

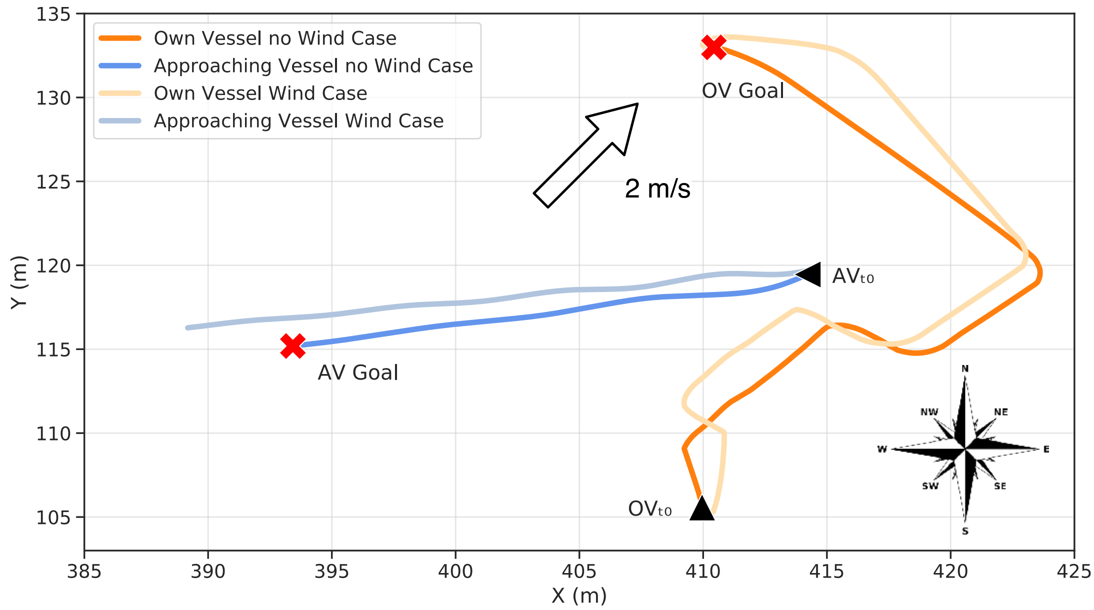


Figure 5.10: Comparison between the trajectories of the vessels in Wind case and no Wind case in a crossing from the right encounter scenario, both cases use ATC A*. Start position for OV and AV are marked with "▲" and "◄", while their goals are marked with "x". For the Wind case, the direction of the wind is northeast, represented by an arrow. We can observe that our ATC A* kept implying COLREGS compliance even under the influence of wind in this scenario.

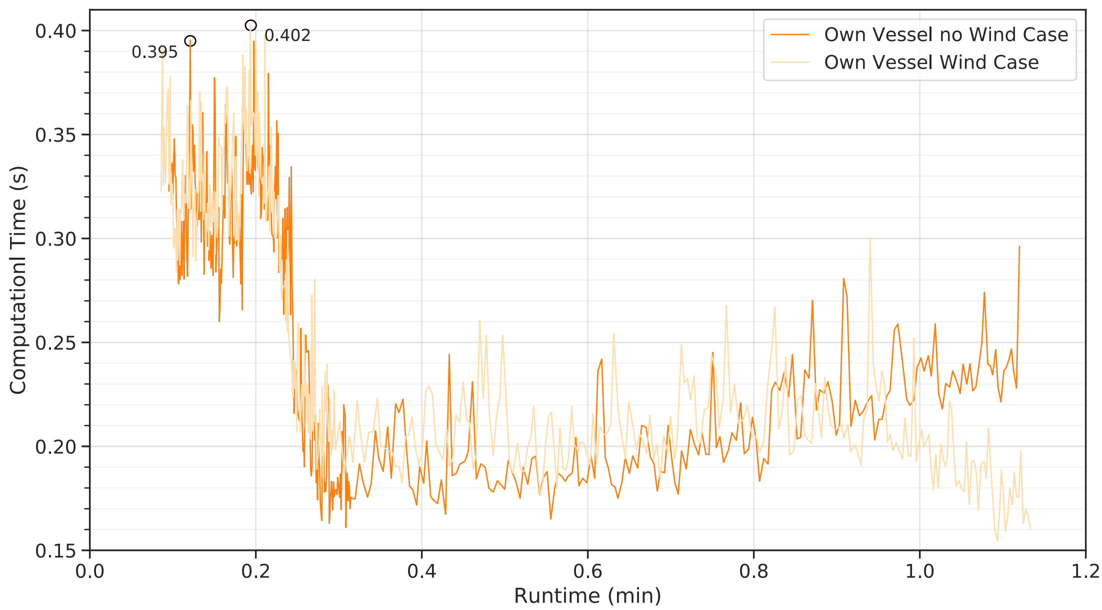


Figure 5.11: Computational time comparison between Wind case versus no Wind in a crossing from the right encounter. The system achieves a peak cost of around 0.395s using ATC without wind and around 0.402s with wind.

In Figure 5.10 we compare final trajectories for the same crossing from the right scenario described in 5.3, now for two executions of the simulation with different wind influence, in both we use ATC A*. "No Wind case" shows the behavior of the OV being influenced by no wind. "Wind case" shows the trajectory of our OV being influenced by the wind with northeast direction and intensity of 2.0 m/s, indicated by an arrow. We can see the change in the trajectory of OV and that it still maintains COLREGS-compliance.

Figure 5.11 shows the computational time measured in seconds over time for wind and no wind cases. We observe that the computational time remains stable and and similar to the previously presented cases. The computational time does not seem to be related to the imposed wind influence for this simulation.

5.5 Crossing from the Left

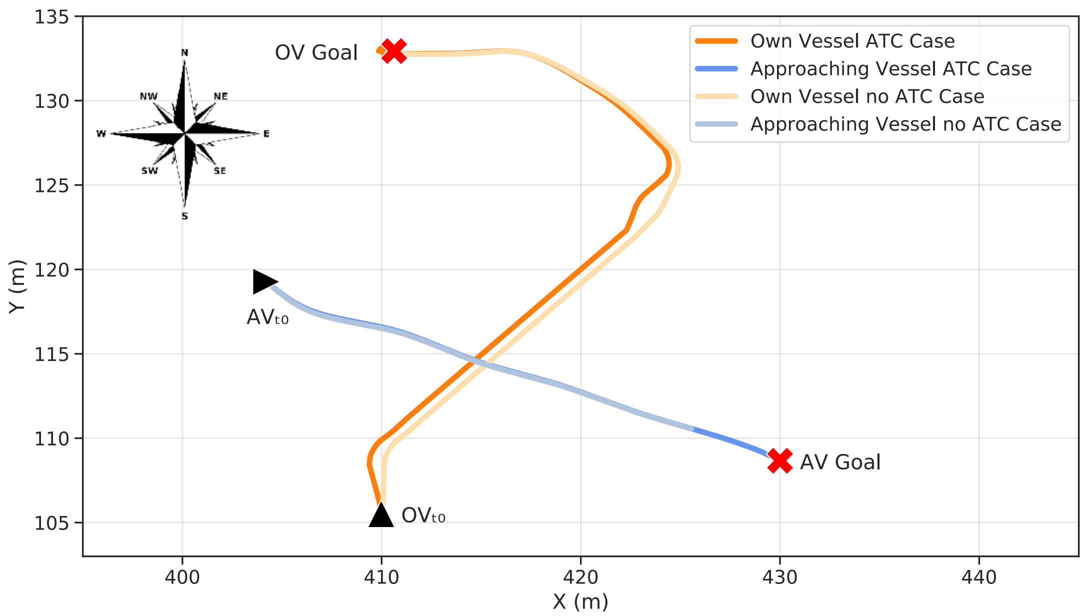


Figure 5.12: Comparison of vessels trajectories in ATC case and no ATC in a crossing from the left encounter scenario. Start position for OV and AV are marked with "▲" and "►", while their goals are marked with "x". In blue we show the trajectory done by the AV. In orange we show the trajectory of the OV with our path planner. In yellow we show the trajectory of the OV without our path planner.

In Figure 5.12, we present the behavior of our system when the OV encounter another vessel coming from the left side. In Figure 5.12, we show the comparison between trajectories with and without ATC. In this situation, both systems perform similar behavior in both scenarios, once for this situation, the OV is not responsible for collision avoidance.

Even so, when AV appears in the OV's local cost map, the OV starts to avoid proximity with AV and go towards its starboard side (right).

Figure 5.13 shows the computational time measured in seconds over time. Our COLREGS-Compliant ATC A* method had a peak cost of 0.388s for path generation for this crossing from the right scenario. As we can see, both ATC and no ATC cases have similar computational cost curves. The fall we see in the collected data (*e.g.,* between 0.6 and 0.8 minutes in ATC case) happens due to a missing valid A* goal. This happens due to error on conversion between global and local locations using ROS move_base standard conversion function.

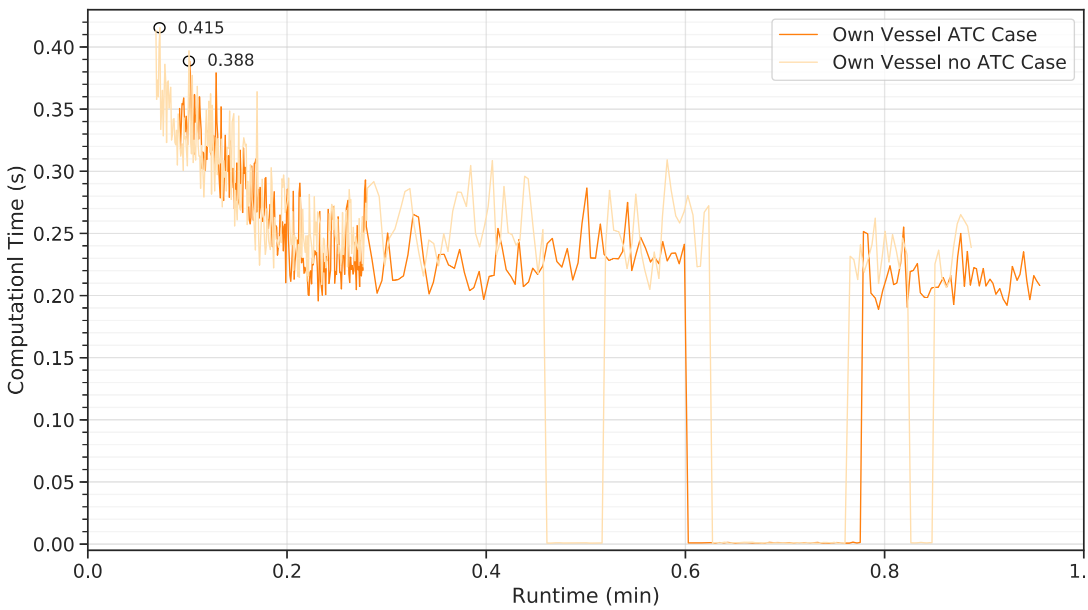


Figure 5.13: Computational time comparison between ATC case versus no ATC in a crossing from the left encounter over time. The system achieves a peak cost of around 0.388s using ATC and around 0.415s without ATC

In Figure 5.14 we compare final trajectories for the same crossing from the left scenario described in 5.3, now for two executions of the simulation with different wind influence, in both we use ATC A*. "No Wind case" shows the behavior of the OV being influenced by no wind. "Wind case" shows the trajectory of our OV being influenced by the wind with northeast direction and intensity of 2.0 m/s, indicated by an arrow.

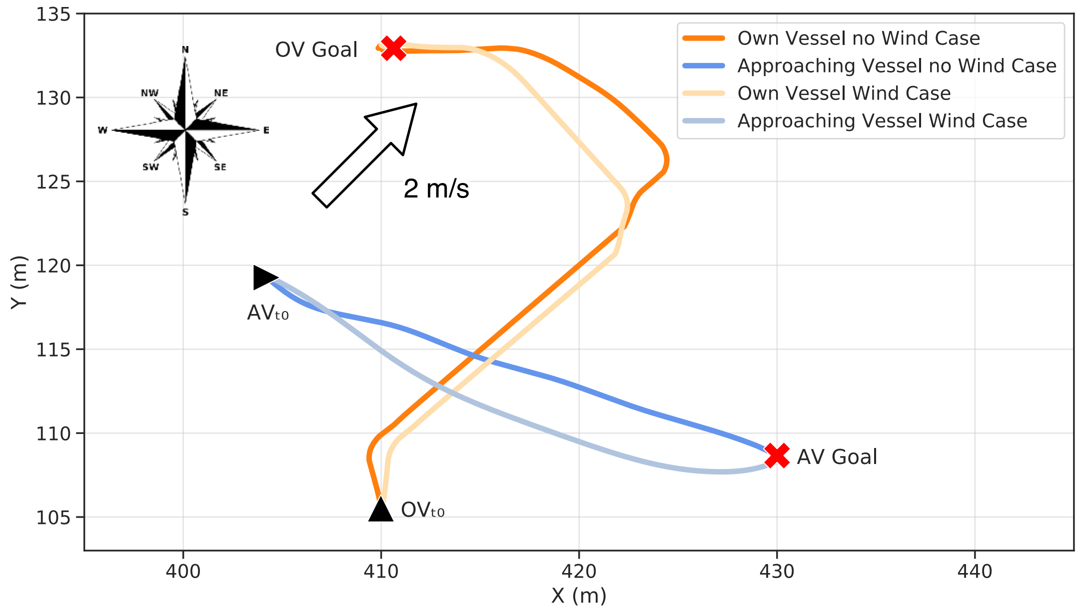


Figure 5.14: Comparison between the trajectories of the vessels in Wind case and no Wind case in a crossing from the left encounter scenario, both cases use ATC A*. Start position for OV and AV are marked with "▲" and "►", while their goals are marked with "x". For the Wind case, the direction of the wind is northeast, represented by an arrow.

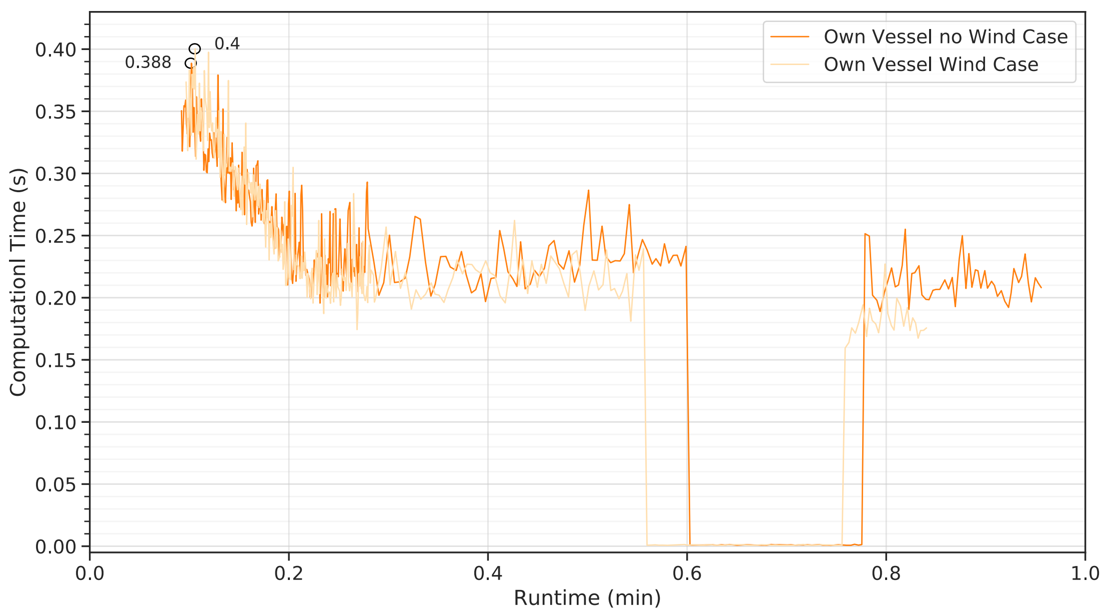


Figure 5.15: Computational time comparison between Wind case versus no Wind in a crossing from the left encounter. The system achieves a peak cost of around 0.388s using ATC without wind and around 0.400s with wind.

Figure 5.15 shows the computational time measured in seconds over time for wind and no wind cases. We observe that the computational time remains stable and similar to the previously presented cases. The computational time does not seem to be related to the imposed wind influence for this simulation. The fall we see in the collected data (*e.g.,* between 0.6 and 0.8 minutes in "ATC case," and between 0.5 and 0.8 minutes in "no ATC case,") happens due to a missing valid A* goal. This happens due to error on conversion between global and local locations using ROS move_base standard function.

5.6 Overtaking

In Figure 5.16, we present the behavior of our system when OV encounters another vessel ahead and decides to overtake it. In Figure 5.16, we show the comparison between trajectories with and without ATC. In this situation, a virtual obstacle was created in the front of the AV and prevent the OV from going in its front. In this situation, both cases ("ATC" and "no ATC") have similar results, since the only restriction in the path planning is related to occupy a position in front of the vessel that is being overtaken (*i.e.,* the AV). The OV avoids going to the front of the AV until the last moment before achieving its goal location. After achieving its goal, the OV receives a new goal from the mission planner.

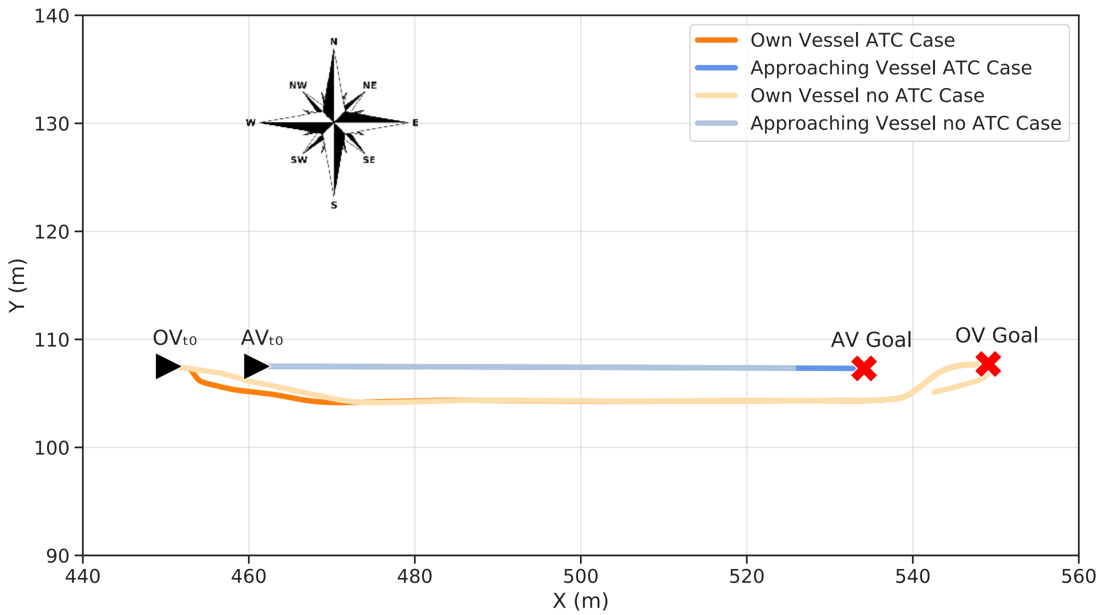


Figure 5.16: Comparison of vessel trajectories in ATC case and no ATC in an overtaking encounter scenario. Start position for OV and AV are marked with "►", while their goals are marked with "x". In blue, we show the trajectory done by the AV. In orange, we show the trajectory of the OV with our path planner. In yellow, we show the trajectory of the OV without our path planner.

Figure 5.17 shows the computational time measured in seconds over time. Our COLREGS-Compliant ATC A* method had a peak cost of 0.356s for path generation for this crossing from the right scenario. As we can see, both ATC and no ATC cases have similar computational cost curves. In Figure 5.17, we can see a high peak of computational time in the beginning, followed by low computational time until the end of this simulation. The high peak happens at the start of the simulation when the AV appears in the OV's local cost map. After the OV's decides to go its starboard side, the path planner can often find a straight path towards the goal, enabling low computational time cost.

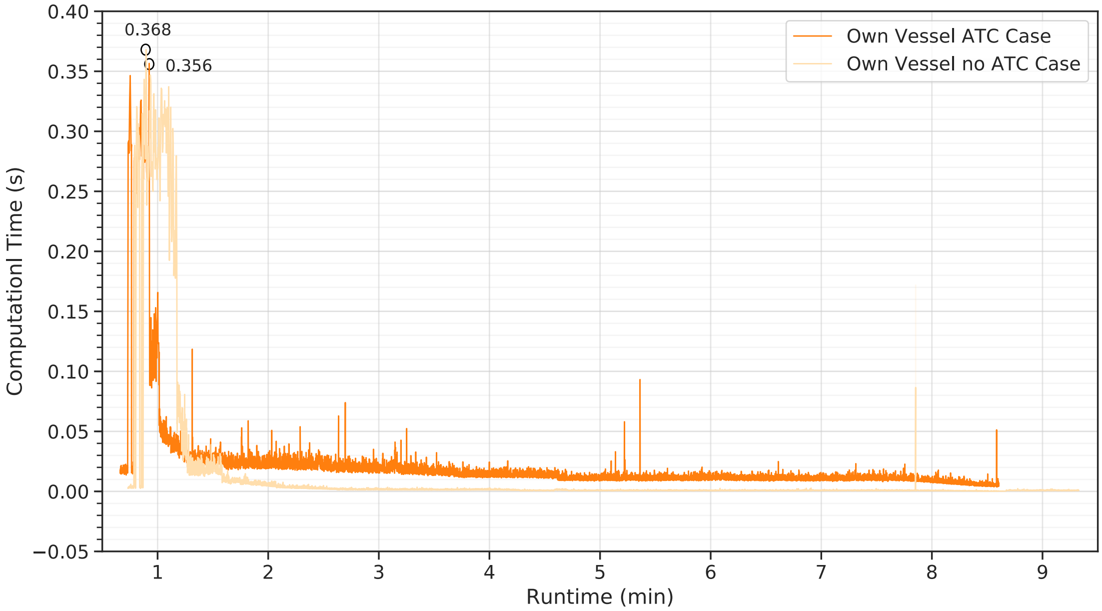


Figure 5.17: Computational time comparison between ATC case versus no ATC in a overtaking encounter over time. The system achieves a peak cost of around 0.356s using ATC and around 0.368s without ATC

In Figure 5.18 we compare final trajectories for the same overtaking scenario described in 5.3, now for two executions of the simulation with different wind influence, in both we use ATC A*. "No Wind case" shows the behavior of the OV being influenced by no wind. "Wind case" shows the trajectory of our OV being influenced by the wind with northeast direction and intensity of 2.0 m/s, indicated by an arrow. In the same way, as in the previously present scenario, the OV was capable of avoiding the collision successfully. The OV avoids going to the front of the AV until the last moment before achieving its goal location. After achieving its goal, the OV receives a new goal from the mission planner.

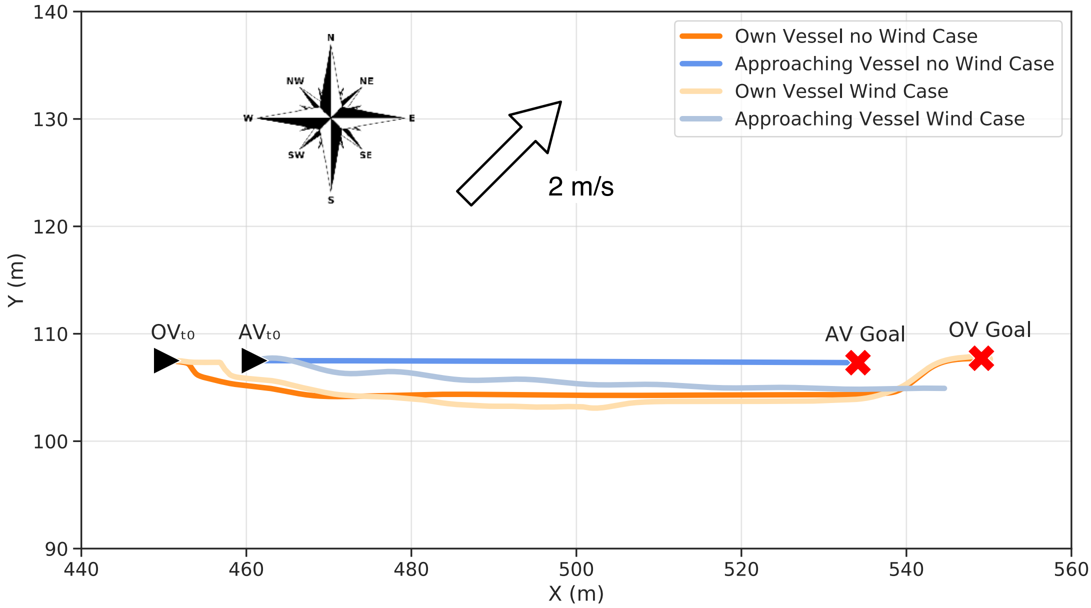


Figure 5.18: Comparison between the trajectories of the vessels in Wind case and no Wind case in an overtaking encounter scenario, both cases use ATC A*. Start position for OV and AV are marked "▶", while their goals are marked with "x". For the Wind case, the direction of the wind is northeast, represented by an arrow.

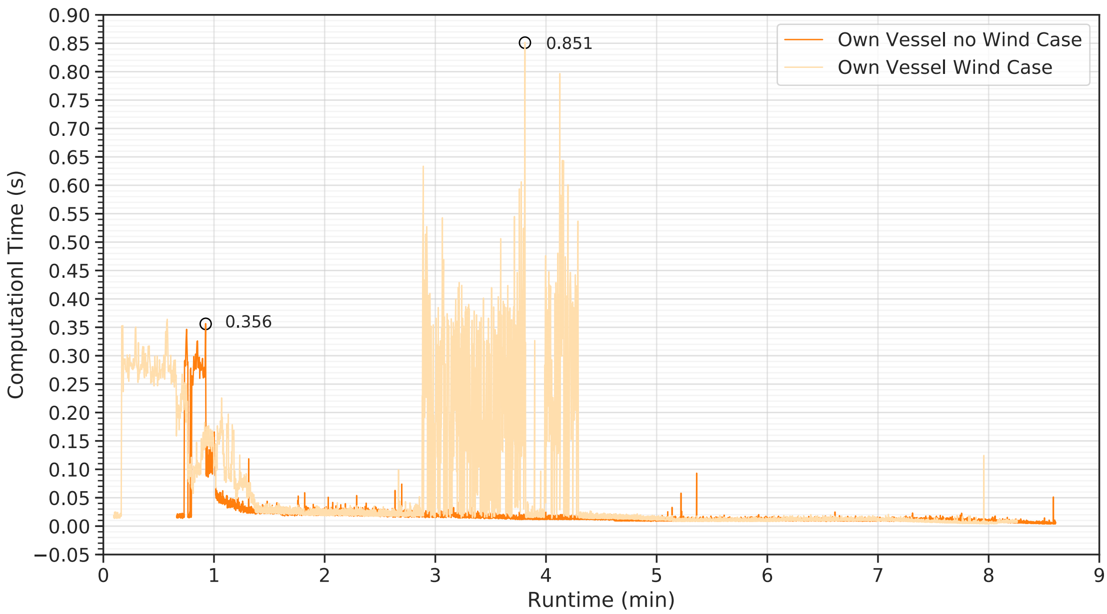


Figure 5.19: Computational time comparison between Wind case versus no Wind in an overtaking encounter. The system achieved a peak cost of around 0.356s using ATC without wind and around 0.851s with wind.

Figure 5.19 shows the computational time measured in seconds over time for wind and no wind cases. In this simulation, we achieved a higher computational time peak. This situation happened, when disturbed by the wind, the AV occupied positions that generate an overlapped position between the local A* goal and the created virtual obstacle (time interval from around 2.85 to 4.4 minutes, in Figure 5.19). In this situation, our system searches for another valid local goal considering the global path within the local cost map. However, once again, in often consecutive attempts, the local planner was unable to found a valid A* goal. Each of these faults triggered the use of the last valid velocity command generated in our system.

5.7 Results Summary

In Table 5.4, we summarize the results for the different simulations we have done. In simulation scenarios, we can see that the computational time peak stays in a range from 0.368 to 0.416 for "no ATC" configuration, from 0.357 to 0.390 for "ATC" configuration and from 0.355 to 0.8852 for "Wind" configuration. Computation time for no ATC and ATC are similar since our ATC method has a low impact on the A* implementation. The influence of wind in the overtaking scenario led our system to an exhaustive search and highest computational time cost peak.

The average time stays similar for each scenario, regardless of the configuration (no ATC, ATC, and Wind) for head-on, crossing from the right, and crossing from the left scenarios. While for the overtaking scenario, the computational time achieved low values due to a simple search ahead for the path planning but presented the highest computational a strong restriction in the search space, caused by an overlap between the created virtual obstacle and the local A* goal.

In all cases presented, there is at least a 1.59[4] ratio between the maximum and average time, implying stability below the maximum computational time sampled. The high values reached occur when there is some considerable restriction in the search space, that is when the goal for the local A* conflicts with the current position of AV or with the created virtual obstacle. For each scenario, we measured the minimum distance kept by the OV from the AV. In our simulations, our system kept at least a distance of 1.505 meters from AV (see Table 5.4).

[4] Ratio between maximum and average computational time for crossing from the left (worst measured ratio)

Table 5.4: Computational time, minimum distance and successfully avoidance for each variation we evaluated regarding the four main encounters described in the COLREGS.

Encounter Type	**Case**	**Computational Time (s)**			**Successful Avoidance**	**Minimum Distance (m)**
		Maximum	**Average**	**Std. Variation**		
Head-On	**No ATC**	0.369	0.136	0.061	Yes	4.431
	ATC	0.364	0.129	0.056	Yes	1.599
	Wind	0.355	0.131	0.060	Yes	1.505
Crossing from Right	**No ATC**	0.390	0.178	0.107	Yes	5.414
	ATC	0.390	0.249	0.060	Yes	3.264
	Wind	0.403	0.253	0.062	Yes	3.739
Crossing from Left	**No ATC**	0.416	0.235	0.105	Yes	7.018
	ATC	0.389	0.224	0.091	Yes	6.274
	Wind	0.400	0.208	0.102	Yes	5.707
Overtaking	**No ATC**	0.368	0.006	0.030	Yes	3.325
	ATC	0.357	0.018	0.022	Yes	3.101
	Wind	0.852	0.039	0.073	Yes	1.787

In table 5.5, we present some comparison parameters between our solution (Jurak) and the solutions of two other authors (Agrawal *et al.* and Huang *et al.*). We chose Agrawal *et al.* for being the work we used as inspiration for our solution and Huang *et al.* for using a vessel of length close to ours. Both studies presented only the achieved results regarding the head-on encounter. Regarding computational time, Agrawal *et al.* present a system with better performance than ours, while Huang *et al.* do not accurately describe the execution time of his solution for collision avoidance. The paper only states that each execution occurs in less than 1 second.

Regarding the minimum distance kept for the head-on encounter, in our simulations, the OV kept from the AV a distance at least 1.6 times bigger than Huang *et al.*. Unfortunately, Agrawal *et al.* do not inform any minimal distance kept. Regarding the OV ‘s maximum achieved speed, our vessel achieved a maximum velocity 2.12 times slower than Huang *et al.*‘s vessel. The maximum achieved velocity is one the main limitation in our solution, since the vessel we use is capable of achieving 1.8 m/s.

Table 5.5: Results Comparison

Comparison Parameter	**Jurak**	**Agrawal** *et al.* [3]	**Huang** *et al.* [20]
Method	ATC A*	A*	VO
Simulator	USV_sim	Matlab	Matlab
CPU	Intel i7 3.4 GHz 16 GB RAM	Intel i5 2.4 GHz	Not Informed
Search Space Dimension	100x100	100x100	N/A
Maximum Computational Time (s)	0.364	0.238	< 1
Vessel Length (m)	1.2	4	1.255
Minimal Distance Kept (m)	1.599	Not Informed	0.992
Own Vessel Max. Speed (m/s)	0.47	Not Informed	1

6. CONCLUSION

In this chapter we discuss the results and future work.

6.1 Results Discussion

In this work, we present a Guidance, Navigation and Control (GNC) system for autonomous Unmanned Surface Vehicles (USVs). Due to the need for vessels that navigate in the water surface to respect Convention on the International Regulations for Preventing Collisions at Sea (COLREGS), we apply a method of path planning to our system to avoid violating COLREGS when a vessel using our system encounters another.

The system we develop is composed of navigation, control, and guidance modules. The navigation system is responsible for perceiving the state of the surrounding environment and the vessel's state itself. Due the fact that obstacle detection at sea is to complex itself, in this work we use laser detection instead of image detection with camera . The control system can modify the state of the USV and move it. The guidance system define, s a path to achieve a goal using the information collected by the navigation system considering the state of the environment and the vessel itself.

Our main contribution was the development and integration of these modules and the adaptation of a technique presented in the literature to make the behavior of a vessel guided by our system respect the COLREGS. Our system can react following the COLREGS when it finds only one approaching vessel. When finding multiple vessels, our system can generate routes to avoid collision, but the COLREGS compliance capacity has not been guaranteed. Another limitation is related to our system, considering that the approaching vessel is of the same type as ours since different categories of vessels imply a change in the COLREGS interpretation.

Our solution uses A* to find a path that leads towards the location goal. When a vessel approaches ours, our local planning module reacts and generates COLREGS-compliant routes. To generate COLREGS compliant routes, we have adapted the solution presented by Agrawal *et al.* [3] in our system. Once Agrawal *et al.* [3] used A* as path finding method, and our approach consists on using Agrawal *et al.* [3] solution, we implemented A* as path finding in our local planner. When our system detects another vessel in its proximity, it creates a virtual obstacle that restricts the search space of our local planner, excluding positions that would violate COLREGS. Thus the local planner is forced to choose a COLREGS-compliant route.

To correctly evaluate the behavior of the system considering two aspects of our interest: COLREGS-compliance and response to wind, we chose to perform the tests separately, that is, we first compare the behavior of the system when using our method, and without using our method, to validate that our method imposes COLREGS-compliance. In the second round of tests, we evaluated the response of the system when under the influence of wind; that is, we compared identical scenarios, one with wind and another without wind. In both scenarios, the system uses our COLREGS-compliant method. Due to the time we had to finish our work, we evaluated only the influence of wind and did not evaluate the influence of waves and water current.

We evaluate the performance of our system regarding the main scenarios of encounter described in COLREGS, they are head-on, crossing from the right, crossing from the left, and overtaking. We collected the maximum and average computational time for the execution of our Artificial Terrain Costs (ATC) A* method in each cycle. For all simulated scenarios, our system was able to avoid collision and follow COLREGS even with a wind intensity of 2 m/s.

Our ATC A* method has been effective for COLREGS-compliance collision avoidance in the scenarios we simulated. Regarding performance, the computational cost is mainly related to our A* implementation. Our A* implementation has several conversions operations between the global and local scope. These operations consist of operations that evaluate the value of each cell in an mxm grid, where m is equal to 100.

The creation of virtual obstacles with ATC, creates indirectly and intermittently an increase in computational time, due to the restriction in the search space. The impact of creating obstacles with ATC is low compared to the total cost since the implementation of ATC for creating virtual obstacles consists of filling in the values on the local cost map in a wxl area, where w is the local width of our vessel and l is the distance between the approaching vessel and the corresponding edge of the local cost map. In the worst case, l is 99, since the side of the cost map location is 100, and the AV must occupy at least one position of the cost map location. Intermittent behavior occurs because, when an obstacle is created and occupies the same position as the local goal, the local planner extinguishes the entire search area and look backward in the global plan for a valid position goal in the local cost maps.

6.2 Future Work

The main improvements are related to the reduction of computational time, an increase in the maximum speed of the vessel, and a greater variety in possible encounters. Regarding computational time, it is necessary to investigate our implementation of A* to find out if there is any possible improvement that impacts on the total time of execution of our local planner. Another major limitation related to our local planner is the reduced ability to generate velocity commands that allow USV to navigate with more than 0.4 m/s. The simulated USV model we used in our tests can navigate up to 1.8 m/s. This improvement requires investigation to understand better where the speed restriction is.

Regarding variety in the encounter, our system has the potential to be simulated in an environment with multiple boats. It is necessary to study ways to apply our method based on the creation of virtual obstacles for situations where simultaneous encounters between multiple vessels occur. Perhaps a necessary change is the removal of lethality from virtual obstacles. Currently, the virtual obstacles created are considered impassable by the local planner. In multiple encounters, there is a potential increase in space restriction, where it is not possible to navigate. Therefore, high-cost virtual obstacles could be created for the planner but with the possibility of transposition if it is the only choice to avoid collision.

Related to the encounters variety, we should study ways to consider different categories of vessels found and take this into account when creating virtual obstacles. For example, where our vessel is power-driven and finds a sailing vessel to its left, instead of keeping its course and waiting for the vessel on left to avoid collision, our vessel assumes the responsibility of evading the collision.

Another possible improvement would be to add a ray of cost inflation around the virtual obstacles created. This could prevent the A* method from exploring locations that are neighbors of virtual obstacles, possibly decreasing the computational cost. In the tests performed, the method explored large areas close to the virtual obstacles, due to the heuristic behavior of the algorithm. However, these obstacles could not be overcome, so these nodes were explored but did not lead to the final trajectory.

Regarding real-world tests, we could try to bound the maximum computational time reducing conversions from the local and global planners. So, we could evaluate the performance of our system when running in the embedded computers we have in the vessels of our laboratory. For real-world tests would be necessary to study how to adequate each of the modules of our system to the hardware available in our vessel.

REFERENCES

[1] Abdelaal, M.; Fränzle, M.; Hahn, A. “Nonlinear model predictive control for trajectory tracking and collision avoidance of underactuated vessels with disturbances”, *Ocean Engineering*, vol. 160, Jul 2018, pp. 168–180.

[2] Abdelaal, M.; Hahn, A. “Nmpc-based trajectory tracking and collision avoidance of unmanned surface vessels with rule-based colregs confinement”. In: Proceedings of the IEEE Conference on Systems, Process and Control (ICSPC), 2016, pp. 23–28.

[3] Agrawal, P.; Dolan, J. M. “Colregs-compliant target following for an unmanned surface vehicle in dynamic environments”. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 1065–1070.

[4] Annamalai, A.; Motwani, A. “A comparison between lqg and mpc autopilots for inclusion in a navigation, guidance and control system”, *Marine and Industrial Dynamic Analysis, School of Marine Science and Engineering, Plymouth University*, vol. 1, Mar 2013, pp. 1–19.

[5] Benjamin, M.; Curcio, J. “COLREGS-based navigation of autonomous marine vehicles”, *IEEE/OES Autonomous Underwater Vehicles*, vol. 1–1, Jun 2004, pp. 32–39.

[6] Benjamin, M. R.; John J. Leonard and Joseph A. Curcio; Newman, P. M. “A Method for Protocol-Based Collision Avoidance between Autonomous Marine Surface Craft”, *Journal of Field Robotics*, vol. 71–5, May 2006, pp. 486–494.

[7] Borenstein, J.; Koren, Y. “Real-time obstacle avoidance for fast mobile robots”, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19–5, Sep-Oct 1989, pp. 1179–1187.

[8] Caccia, M.; Bibuli, M.; Bono, R.; Bruzzone, G.; Bruzzone, G.; Spirandelli, E. “Unmanned surface vehicle for coastal and protected waters applications: the charlie project”, *Marine Technology Society Journal*, vol. 41–2, Jun 2007, pp. 62–71.

[9] Caccia, M.; Bono, R.; Bruzzone, G.; Bruzzone, G.; Spirandelli, E.; Veruggio, G.; Stortini, A. M.; Capodaglio, G. “Sampling sea surfaces with SESAMO: An autonomous craft for the study of sea-air interactions”, *IEEE Robotics and Automation Magazine*, vol. 12–3, Sep 2005, pp. 95–105.

[10] Campbell, S.; Abu-Tair, M.; Naeem, W. “An automatic COLREGs-compliant obstacle avoidance system for an unmanned surface vehicle”, *Proceedings of the Institution of Mechanical Engineers Part M: Journal of Engineering for the Maritime Environment*, vol. 228–2, May 2013, pp. 108–121.

[11] Campbell, S.; Naeem, W.; Irwin, G. W. “A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres”, *Annual Reviews in Control*, vol. 36–2, Dec 2012, pp. 267–283.

[12] Candeloro, M.; Lekkas, A. M.; Sørensen, A. J. “A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels”, *Control Engineering Practice*, vol. 61–5, Apr 2017, pp. 41–54.

[13] Damasio, J.; Jurak, D.; Bittencourt, R.; de Borba Campos, M.; Amory, A. “Programming teaching with robotic support for people who are visually impaired: a systematic review”. In: Proceedings of the Brazilian Symposium on Computers in Education (SBIE), 2019, pp. 1231–1235.

[14] Domingues, A. R.; Jurak, D. A.; Sergio Filho, J.; Amory, A. d. M. “Integrating an mpsoc to a robotics environment”. In: Proceedings of the Latin American Robotics Symposium (LARS), Brazilian Symposium on Robotics (SBR) and Workshop on Robotics in Education (WRE), 2019, pp. 204–209.

[15] Fiorini, P.; Shiller, Z. “Motion planning in dynamic environments using velocity obstacles”, *International Journal of Robotics Research*, vol. 17–7, Jul 1998, pp. 760–772.

[16] Group, O. M. R. “Mission oriented operating suite”. Source: https://ori.ox.ac.uk/moos/, March 2020.

[17] He, Y.; Jin, Y.; Huang, L.; Xiong, Y.; Chen, P.; Mou, J. “Quantitative analysis of colreg rules and seamanship for autonomous collision avoidance at open sea”, *Ocean Engineering*, vol. 140, Jul 2017, pp. 281–291.

[18] Healey, A. J.; Horner, D. P.; Kragelund, S. P.; Wring, B.; Monarrez, A. “Collaborative unmanned systems for maritime and port security operations”, *IFAC Proceedings Volumes*, vol. 7–PART 1, Jul 2007, pp. 1–6.

[19] Healey, A. J.; Lienard, D. “Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles”, *IEEE Journal of Oceanic Engineering*, vol. 18–3, Jul 1993, pp. 327–339.

[20] Huang, Y.; Chen, L.; van Gelder, P. “Generalized velocity obstacle algorithm for preventing ship collisions at sea”, *Ocean Engineering*, vol. 173–1, Feb 2019, pp. 142–156.

[21] IMO. “Convention on the international regulations for preventing collisions at sea, 1972”. Source: http://www.imo.org/en/About/Conventions/ListOfConventions/Pages/COLREG.aspx, March 2020.

[22] Jian Yan, R.; Pang, S.; Bing Sun, H.; Jie Pang, Y. “Development and missions of unmanned surface vehicle”, *Journal of Marine Science and Application*, vol. 9–4, Dec 2010, pp. 451–457.

[23] Jorge, V. A.; Granada, R.; Maidana, R. G.; Jurak, D. A.; Heck, G.; Negreiros, A. P.; Dos Santos, D. H.; Gonçalves, L. M.; Amory, A. M. “A survey on unmanned surface vehicles for disaster robotics: Main challenges and directions”, *Sensors*, vol. 19–3, Jan 2019, pp. 702–705.

[24] Kiencke, U.; Nielsen, L.; Sutton, R.; Schilling, K.; Papageorgiou, M.; Asama, H. “The impact of automatic control on recent developments in transportation and vehicle systems: Status report prepared by the ifac coordinating committee on transportation and vehicle systems”, *IFAC Proceedings Volumes*, vol. 38, Jan 2005, pp. 109–116.

[25] Kuwata, Y.; Wolf, M. T.; Zarzhitsky, D.; Huntsberger, T. L. “Safe maritime autonomous navigation with COLREGS, using velocity obstacles”, *IEEE Journal of Oceanic Engineering*, vol. 39–1, May 2014, pp. 110–119.

[26] Larson, J.; Bruch, M.; Ebken, J. “Autonomous navigation and obstacle avoidance for unmanned surface vehicles”. In: Proceedings of the Unmanned Systems Technology (UST), 2006, pp. 201–207.

[27] Larson, J.; Bruch, M.; Halterman, R.; Rogers, J.; Webster, R. “Advances in Autonomous Obstacle Avoidance for Unmanned Surface Vehicles”, Technical Report, Space and Naval Warfare Systems Center, 2007, 15p.

[28] Lee, S.-M.; Kwon, K.-Y.; ; Joh, J. “A fuzzy logic for autonomous navigation of marine vehicle satisfying COLREG guidelines”, *International Journal of Control, Automation and Systems*, vol. 2–2, Jun 2004, pp. 171–181.

[29] Liu, Z.; Zhang, Y.; Yu, X.; Yuan, C. “Unmanned surface vehicles: An overview of developments and challenges”, *Annual Reviews in Control*, vol. 41, Jan 2016, pp. 71–93.

[30] LLC, P. “Platypus llc.” Source: http://senseplatypus.com/, March 2020.

[31] Loe, Ø. “Collision avoidance concepts for marine surface craft”, Master thesis, Norwegian University of Science and Technology, 2007, 111p.

[32] Naeem, W.; Irwin, G. W. “Evasive Decision Making in Uninhabited Maritime Vehicles”, *IFAC Proceedings Volumes*, vol. 44–1, Jan 2011, pp. 12833–12838.

[33] Naeem, W.; Irwin, G. W.; Yang, A. “COLREGs-based collision avoidance strategies for unmanned surface vehicles”, *Mechatronics*, vol. 22–6, Sep 2012, pp. 669–678.

[34] Naus, K.; Wąż, M. “The idea of using the A* algorithm for route planning an unmanned vehicle “Edredon””, *Zeszyty Naukowe / Akademia Morska w Szczecinie*, vol. 36, Jun 2013, pp. 143—-147.

[35] Paravisi, M.; de Morais Amory, A.; et al.. “Toward accurate hydrologic urban flooding simulations for disaster robotics”. In: Proceedings of the International Conference on Informatics in Control, Automation and Robotics (ICINCO)., 2018, pp. 6–12.

[36] Paravisi, M.; H. Santos, D.; Jorge, V.; Heck, G.; Gonçalves, L.; Amory, A. “Unmanned surface vehicle simulator with realistic environmental disturbances”, *Sensors*, vol. 19–5, Mar 2019, pp. 1068–1068.

[37] Pastore, T.; Djapic, V. “Improving autonomy and control of autonomous surface vehicles in port protection and mine countermeasure scenarios”, *Journal of Field Robotics*, vol. 27–6, Nov 2010, pp. 903–914.

[38] Pin, F. G.; Vasseur, H. A. “Autonomous trajectory generation for mobile robots with non-holonomic and steering angle constraints”, Technical Report, Oak Ridge National Lab., 1990, 102–103p.

[39] Quigley, M.; Conley, K.; Gerkey, B. P.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A. Y. “Ros: an open-source robot operating system”. In: Proceedings of the ICRA Workshop on Open Source Software (IWOSS), 2009, pp. 65–69.

[40] Rong Li, X.; Jilkov, V. P. “Survey of maneuvering target tracking. part i. dynamic models”, *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39–4, Oct 2003, pp. 1333–1364.

[41] Russell, S. J.; Norvig, P. “Artificial intelligence”. Prentice Hall/Pearson Education, 2003, 1152p.

[42] Shanghai Slamtec Co., L. “Rplidar a3”. Source: http://www.slamtec.com/en/Lidar/A3, March 2020.

[43] Slabaugh, G. G. “Computing euler angles from a rotation matrix”, *Retrieved on August*, vol. 6–2000, Aug 1999, pp. 39–63.

[44] Soltan, R. A.; Ashrafiuon, H.; Muske, K. R. “TRAJECTORY REAL-TIME OBSTACLE AVOIDANCE FOR UNDERACTUATED UNMANNED SURFACE VESSELS”. In: Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (AIDET), 2009, pp. 1–9.

[45] Svec, P.; Gupta, S. K. “Automated planning logic synthesis for autonomous unmanned vehicles in competitive environments with deceptive adversaries”, *Studies in Computational Intelligence*, vol. 341, Feb 2011, pp. 171–193.

[46] Svec, P.; Gupta, S. K. “Automated synthesis of action selection policies for unmanned vehicles operating in adverse environments”, *Autonomous Robots*, vol. 32–2, Jul 2012, pp. 149–164.

[47] Svec, P.; Schwartz, M.; Thakur, A.; Gupta, S. K. “Trajectory planning with look-ahead for unmanned sea surface vehicles to handle environmental disturbances”. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2011, pp. 1154–1159.

[48] Xu, Q.; Zhang, C.; Zhang, L. “Deep convolutional neural network based unmanned surface vehicle maneuvering”. In: Proceedings of the Chinese Automation Congress (CAC), 2017, pp. 878–881.

[49] Yang, A.; Niu, Q.; Zhao, W.; Li, K.; Irwin, G. W. “An efficient algorithm for grid-based robotic path planning based on priority sorting of direction vectors”, *Lecture Notes in Computer Science*, vol. 63–PART 2, Sep 2010, pp. 456–466.

[50] Zhuang, J. Y.; Su, Y. M.; Liao, Y. L.; Sun, H. B. “Motion planning of USV based on Marine rules”, *Procedia Engineering*, vol. 15, Jan 2011, pp. 269–276.