Autonomous Launch and Recovery of small unmanned underwater vehicles

Author :
Mr Aurélien Grenier - ROB CI2020

Tutors :
Mr Karl Sammut
Mr Benoit Clément

ENSTA BRETAGNE

NICOLAS BAUDIN
INTERNSHIPS IN FRANCE INITIATIVE

Flinders UNIVERSITY

Acknowledgment

I would like to particularly thank my tutors Mr Karl Sammut and Mr Benoit Clément who accompanied me throughout this training course and allowed me to supervise its progress.

I would also like to thank Mr Jonathan Wheare and Mr Andrew Lammas who were able to provide me with valuable technical assistance in the development of the algorithms.

I would also like to thank the PhD student from ENSTA Bretagne, Joris Tillet, and the former PhD student, Thomas le Mézo, who were available to answer my questions during the lockdown phase but also afterwards.

Summary — The search for underwater objects is an area that is highly invested by the mobile robotics sector these days, which benefits from numerous investments. The X300 robot is an AUV[1] that can acquire sonar data for this type of mission. The main objective of this end-of-studies project is to allow the use of this robot in a swarm of surface and submarine AUVs in order to improve the performances of our AUV. As this project includes different parts, we have chosen to work on the control of these AUVs in a training. The development was done on the Gazebo simulator with the help of the ROS[2] middleware allowing to simulate the internal and external communications of the robots. This report presents the software and theoretical development that resulted in a controller to operate AUV in drone swarms. Several controllers are exposed in this report because they depend on the function of the AUV to which it is associated.

Résumé — La recherche d'objet sous-marins est domaine très investi par le secteur de la robotique mobile de nos jours qui profite de nombreux investissements. Le robot X300 est un AUV permettant d'acquérir des données sonars pour ce type de mission. L'objectif principal de ce projet de fin d'études et de permettre l'utilisation de ce robot dans un essaim de drones en surface et sous-marins afin d'amélioration les performances de notre AUV. Ce projet comprenant différentes parties, nous avons choisi de travailler sur le contrôle de ces AUVs dans une formation. Le développement s'est fait sur le simulateur Gazebo avec l'aide du middleware ROS permettant de simuler les communications internes et externes aux robots. Ce rapport présente le développement logiciel et théorique qui a permis d'en résulter un controleur pour exploiter des AUVs en essaim de drones. Plusieurs controleurs sont exposés dans ce rapport car ils dépendent de la fonction de l'AUV auquel il est associé.

ENSTA BRETAGNE

NICOLAS BAUDIN
INTERNSHIPS IN FRANCE INITIATIVE

Flinders
UNIVERSITY

[1] Autonomous Underwater Vehicle
[2] Robot Operating System

Contents

List of Figures

Introduction

The X300 is an autonomous underwater robot designed by Graal Tech with a torpedo shape. It is a robot with on-board systems for autonomous navigation that can produce sonar images for underwater research and inspection. This system is perfect for shallow depth missions due to its small size. The objective of this project is to build on the hardware architecture to develop a software architecture to carry out the missions mentioned above but in a swarm formation of robots.

The main difficulty of this project lies in the communication between robots in order to allow the regulation of the whole robot group taking into account possible disturbances. It was therefore necessary to define the different roles of each robot within the swarm which will then allow us to develop our regulation algorithms. The interest of the preliminary identification of the roles allows us to anticipate possible collisions between robots in a swarm.

This report aims at detailing the stages of development of such controllers in order to make them stable, precise and fast. A particular attention was conveyed to the theoretical foundations which made it possible to develop these controllers. We will end with a presentation of the possible improvements to the project that we identified during the development of the algorithms.

Chapter 1

Context

1.1 Presentation of the university and the company

This internship was possible within the framework of the "Nicolas Baudin" exchange program. This is a partnership between the Thales Group and Naval Group companies and Australian universities. Thales Group therefore provided financial support for this project carried out at Flinders University in Adelaide. This internship is a continuation of the exchange and cooperation programs between France and Australia in the maritime field.

Flinders University trains their students in many different fields, from medicine to engineering to mathematics. I had the opportunity to work in the marine robotics research group. Because Adelaide is a coastal city, it allows Flinders University to conduct sea trials on their systems. The university has surface and underwater vehicles. We will then present the X300 robot used during the internship.

1.2 Implications

Seabed explorations and inspections are main issues regarding many fields. For mine warfare, oil well research, wreck searching we need information about the seabed. Over several decades of progress, different systems have been devised to obtain this information. Over several decades of progress, different systems have been devised to obtain this information. Although the mapping of the seabed is mainly carried out by surface vehicles, surface and underwater robots are widely used for the production of sonar imagery over small areas. The choice of stand-alone vehicles depends on the situation, but they generally allow access to areas that are more difficult to access compared to deep-draught boats.

In particular, we can differentiate between several families of robots, autonomous or not. The AUVs (Autonomous Underwater Vehicle), ROVs (Remotely Operated underwater Vehicle) and USVs (Unmanned Surface Vehicle). ROVs are more maneuverable and allow simplified and supervised exploration of inaccessible areas. USV are generally used to deploy other robots. These robots are widely used for sonar imaging.

1.3 Aims

The most developed AUVs are those used for deep waters. Their positioning systems (IMU[1], sensors...) are often very expensive but very accurate. On the other hand, due to their large size and the difficulty to deploy them, it is not possible to use them in shallow water. Smaller AUVs such as the X300 are also being developed to meet this need. Being smaller in size, they therefore have smaller batteries, less computer storage and their positioning systems are less efficient. However, they are subject to the same current conditions as other AUVs and are therefore more sensitive to them due to their size. An interesting idea would be to use several AUVs in the same mission so that they collaborate to improve their autonomy, their geolocation and thus the time of accomplishment of the mission. This is therefore the focus of our study.

1.4 Equipment

In order to work on this project, Flinders University has an X300 AUV developed by the company Graal Tech. This model is a small 3 m long, 150 mm diameter AUV with the following sensors and actuators:

- 5 tunnel thrusters to control the robot on 5 degrees of freedom (surge, heave, sway, pitch and yaw), roll control is passive.
- an attitude and heading reference system (AHRS) and a GPS[2] to know the position and orientation of the robot
- a pressure sensor to know the depth
- a wifi module to communicate with the other robots and the main ship while on the surface
- an acoustic modem for underwater communication
- an ultra short baseline (USBL) array
- a doppler velocity logger (DVL) to measure velocity relative to the seabed
- a forward looking multibeam imaging sonar to get information about distances to other objects
- a camera for docking and inspection

1.5 Approach chosen

The main goal of the project is to enable data acquisition missions by swarms of AUVs to be carried out autonomously. This objective can be broken down into several sub-objectives.

[1]Inertial Measurement Unit
[2]Global Positioning System

Indeed, it is possible to separate the development of the autonomous navigation part and the data acquisition part.

The autonomous navigation part can itself be broken down. A first sub-part includes the implementation of AUVs at the place of their mission. Then comes the conduct of the mission and the navigation in swarm of AUVs. And a last part concerning the recovery of the AUV and the data it has acquired.

It is important to specify that this course was conducted entirely by telework because of the health situation due to COVID-19. As this situation left the possibility of working only on a simulator, my work during this course focused on the navigation part in swarms of AUVs. The launch and recovery part being more difficult to simulate and to apprehend if we could not attend a sea trial.

The development of the AUV swarm navigation part has been carried out thanks to the Gazebo simulator, the ROS middleware and the uuv_simulator[6] package from ROS. Gazebo allows us to simulate the mission environment and the physics of the robot while ROS allows us to simulate the communications between the AUV's embedded systems and between the different robots. The uuv_simulator package provides tools for AUV control.

Chapter 2

AUV swarm navigation

2.1 State of the art

Swarm navigation is a sector in full expansion and concerns all areas of robotics, whether terrestrial, air or marine. For example, many light shows[4] are organized thanks to swarms of drones. Research in the military field[9] is also being carried out in order to employ swarms of drones. All of them move in a formation to avoid collision with other robots. These formations can defer according to the use of the swarm. It is thus necessary to know the type of mission in which the drone swarm will be employed.

2.1.1 Mission Type

The AUV X300 from Flinders University can be used for underwater search and inspection missions, so it is capable of producing sonar images. As mentioned above, it is small in size and is therefore used in shallow water. The action range of its side scan sonars is therefore smaller than for larger AUVs. In order to scan an entire area, the AUV usually follows a trajectory called a boustrophedon. This trajectory in figure 2.1 allows the simplest possible decomposition of an area in order to obtain an overlap for the data and to minimize the changes in direction that impact the data.

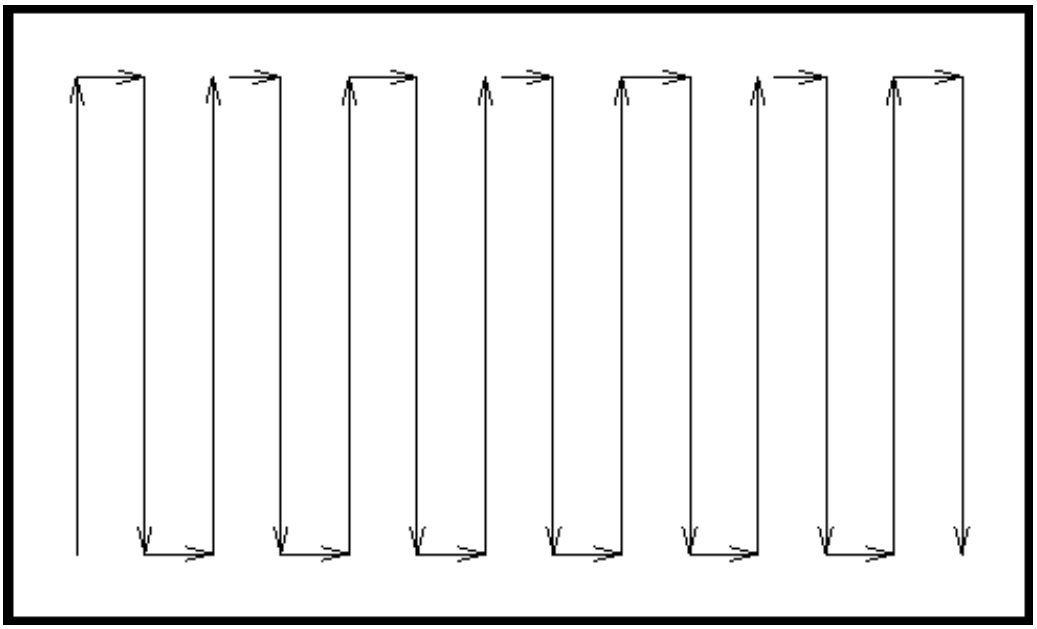


Figure 2.1: Example of boustrophedon trajectory for a rectangular area

During these data acquisition missions, the use of several AUVs makes it possible to cover a larger area simultaneously, thus saving battery power. Small AUVs are more sensitive to current variations which can increase their drift during a mission and thus impact the acquired data. Surface AUVs are therefore needed to help geolocate the diving AUVs which acquire the data.

2.1.2 Formation Type

Diving AUVs

When you look at the shape of a boustrophedon you can see that there is no point in having the AUVs in formation passing through the same position. If they are side by side, they can cover a larger area and reduce the number of turns. Figures 2.2 and 2.3 help to illustrate this issue. By reducing the number of turns, the mission time is reduced. This reduces the amount of data stored per AUV as well as saving energy and reducing the impact of IMU drift on the data.

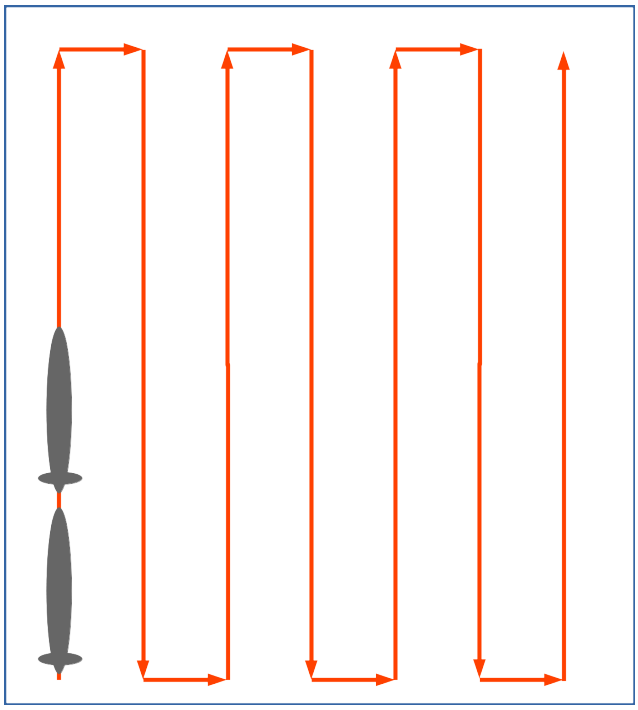


Figure 2.2: Example of 2 AUVs in single line

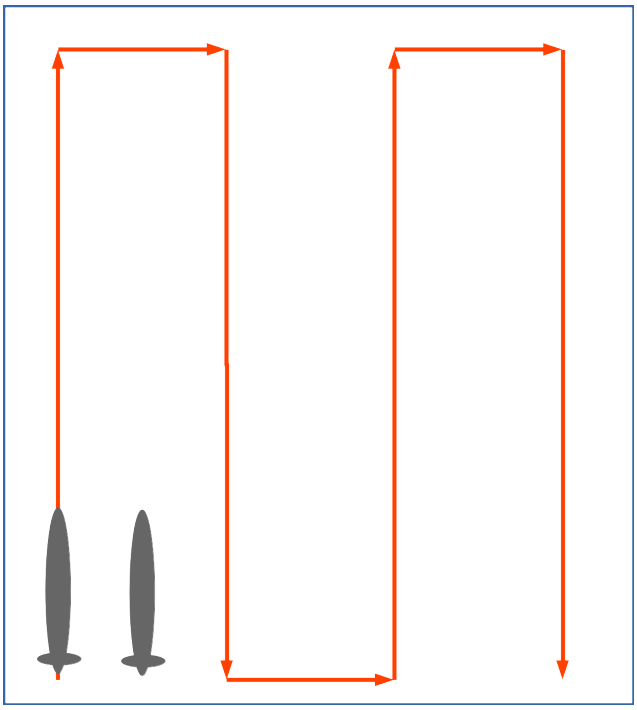


Figure 2.3: Example of 2 AUVs alongside

Surface AUVs

As we said before, robots on the surface can assist the geolocation of diving robots. Using the on-board acoustic modems on the X300, it is possible for the underwater AUVs to ping the surface AUVs to get distance information. With these information, the diving AUVs can compute their position by trilateration. The principle is similar to the satellite positioning system (GNSS[1]) but using robots instead of satellite.

As with satellite navigation, the more robots you have on the surface, the more information you will have about the robot's position. We cannot get information about the orientation of the robot, but it is possible to determine the 3 degrees of freedom in position. In order to determine these three degrees of freedom, we know that we need at least three independent pieces of information. In order to simplify swarm navigation, we fix the altitude of the submerged robots. So theoretically we would need two robots on the surface to determine the last two degrees of freedom.

But when we look at figure 2.4, we can see that two positions are still possible for our robot. Thanks to the altitude we know that the AUV is on he's on a constant altitude plane, our two AUV pings are represented by the two circles. The intersection of two circles in the same plane gives two positions.

[1]Global Navigation Satellite System

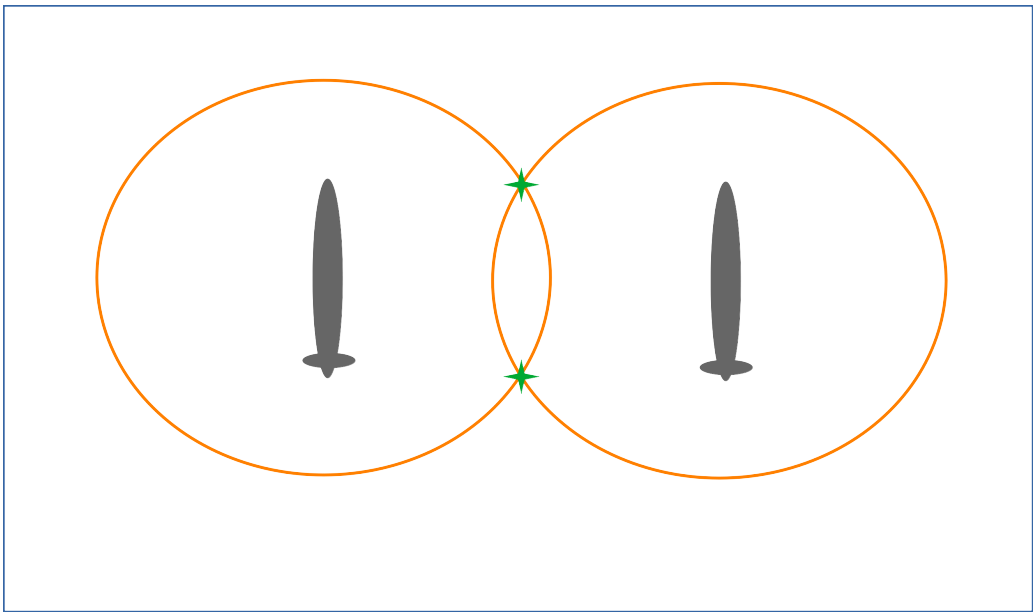


Figure 2.4: Two AUVs at surface

It is therefore necessary to use at least three AUVs as shown in figure 2.5.

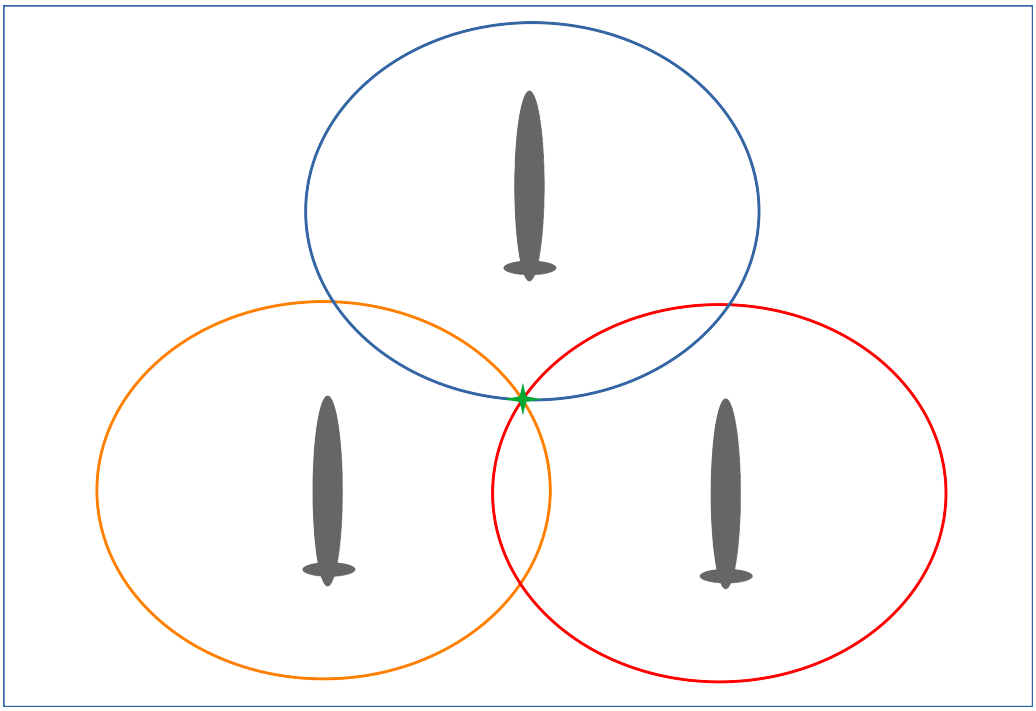


Figure 2.5: Three AUVs at surface

Choosing more AUVs would not be useful in our case, it would increase the possibility of collision without providing additional information. On the other hand, we can only determine the robot's position accurately if the robots' pings on the surface are received at the same time. The accuracy of this system is therefore linked to the frequency of emission of these pings.

Other similar existing projects such as the MONSUN project[3] have developed formation with two surface AUVs for localization. This approach is possible by deducing one of the two possible positions by taking into account the robot dynamics and the last known position of the robot, in particular with a kalman filter. In order to facilitate the implementation of the AUV swarm, we have chosen the formation with three AUVs which allows us to have the maximum of information.

Final formation

Two types of formation can result from our criteria for surface and diving AUVs:

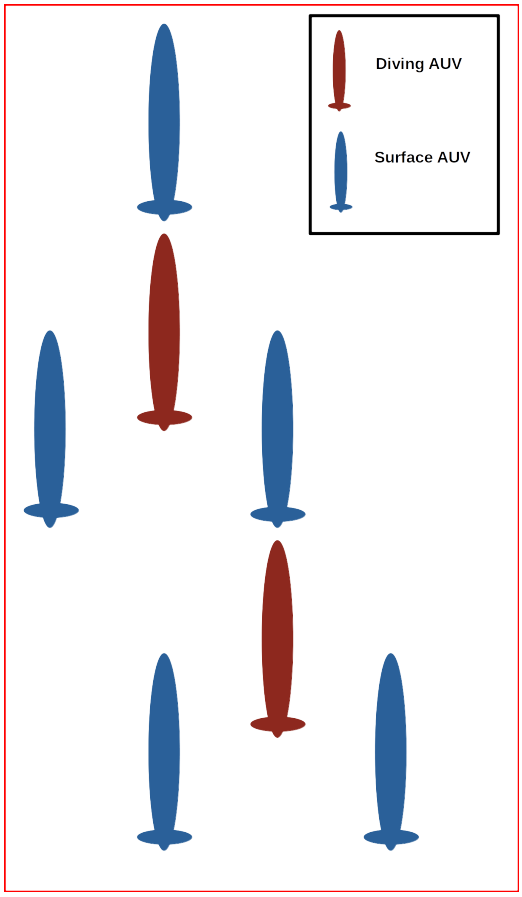


Figure 2.6: Arrow head formation

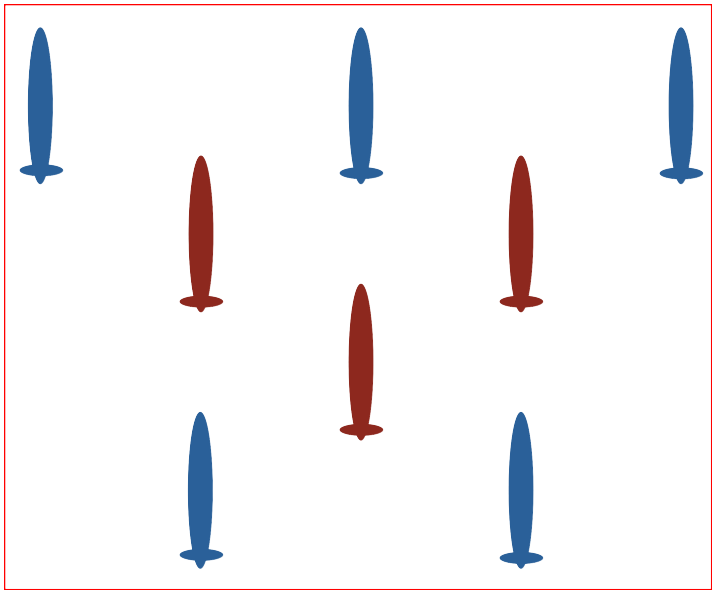


Figure 2.7: Line formation

These two formations differ in size, on figure 2.6 the formation is longer and leaves the second AUV set back from the first. All AUVs are behind a leading AUV in this formation. In the figure 2.7, the submerged AUVs are at the same level, so the formation is more compact. However, not all AUVs are behind a main AUV, this is a point to note that we will talk about later. The choice of formation is rather arbitrary at this stage and depends mainly on the area where the robots will be used. We will see later on the advantages of compact formation, which is not the one chosen in a project such as the AUV MONSUN project[2].

In the context of mine warfare or sonar imaging research, these are planned missions and

therefore the route is predefined. It is therefore necessary in our formation to choose an AUV who knows the mission and the route and who will play the role of leader. This leading robot will have to communicate the positions to the other AUVs to position oneself in formation the formation. The chosen formation will be the arrow-head-shaped one because it lends itself well to this type of management and has been validated by other projects such as the MONSUN project.

2.2 Project architecture

In order to present how this formation has been implemented it is necessary to present the functional architecture of the project. Indeed, there are three different roles in our project for AUVs.

The first AUV that follows the predefined trajectory by waypoints, in parallel, it calculates the positions of the AUVs in the formation and communicates these positions to the other AUVs. It also communicates its own position to the diving AUVs, which allows trilateration.

Concerning the role of the AUVs on the surface. They receive their own position information in the formation. They regulate the system to follow these positions. They must also communicate their own position to the underwater AUVs.

For diving AUVs, their main role is to collect data from other AUVs and ping them for distance information. All of these data allow the diving AUVs to calculate their position. At the same time, they acquire the data.

A diagram in the appendix 3.2.2 illustrates and summarizes these explanations.

2.2.1 Leading AUV

As said before the main AUV must be controlled by planning. For this purpose a list of waypoints is usually used, through which the robot has to run. These waypoints are used to describe the trajectory in boustrophedon, which allows the desired area to be squared.

For this part, we have used the uuv_simulator package which allows to choose a list of waypoints and regulates the system to follow this a trajectory based on these waypoints. Indeed, there are two important features, the thruster manager and the control interface.

The thruster manager allows you to get rid of the robot physics during the development part. Indeed, the thruster manager loads a configuration file of the thrusters which allows it to know the physics of the thrusters. This allows us to have a high level relationship for the regulation by focusing on the forces to be applied on the robot which will be converted by the thruster manager. In our case, for a given force, the manager will convert it into five commands for the motors of the thrusters. It is also possible to apply moments to the thrusters when possible.

If we take a look at the physics of the X300 AUV as in figure 2.8. We see that it is

possible to apply forces on all three axes. The pitch can be altered with a differential of the two opposite vertical thrusters. Similarly, the yaw can be controlled with the two opposite horizontal thrusters. However, neither thruster can change the roll.

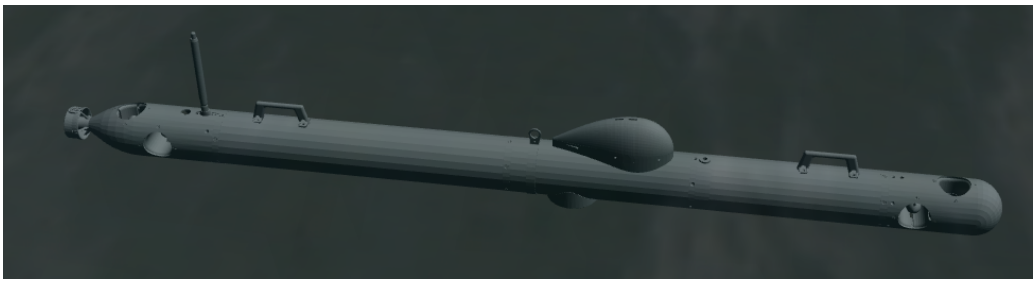


Figure 2.8: X300 in the Gazebo simulator

This manager therefore facilitates the development of a regulator. We will come back to it when explaining the controllers of the other AUVs.

The control interface allows us to load a waypoint file giving the position of the waypoints and the maximum speed of passage at these waypoints. The interface interpolates these waypoints to create intermediate points describing a curve through which the AUV is supposed to run. These intermediate points are called "TrajectoryPoint", it is an information in position, speed and acceleration.

It is possible to choose the type of interpolation of our trajectory, if we want to keep our trajectory as defined by our list of waypoints we just choose a linear interpolation. In our case we have chosen an interpolation with locally curved. This is the linear interpolation with polynomial blend (LIPD[2]). This choice allows a better fluidity in the regulation of the AUV. You can see the difference between these two interpolations in figure 2.9 and 2.10.

[2]Linear Interpolator with polynomial blend

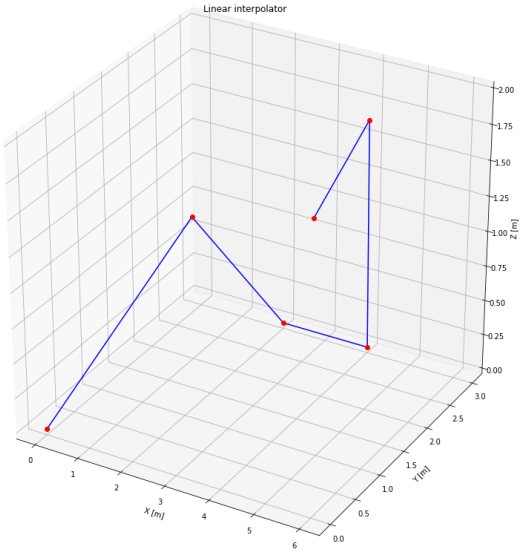


Figure 2.9: Linear interpolation

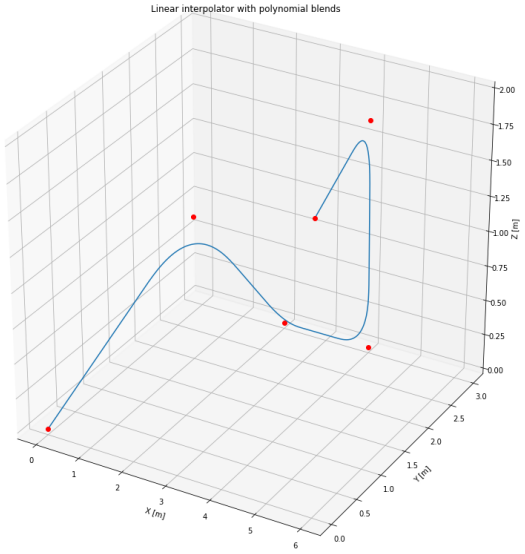


Figure 2.10: Linear interpolation with polynomial blend

There are other types of interpolation but they are not interesting in the case of area scan missions.

For a very simple boustrophedon we can obtain the following figures:

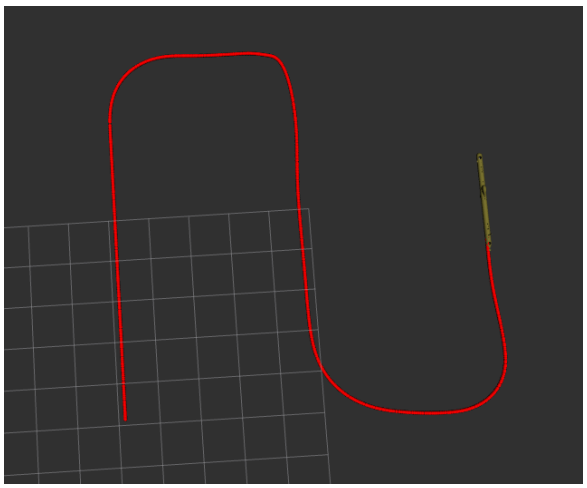


Figure 2.11: Waypoints describing a boustrophedon

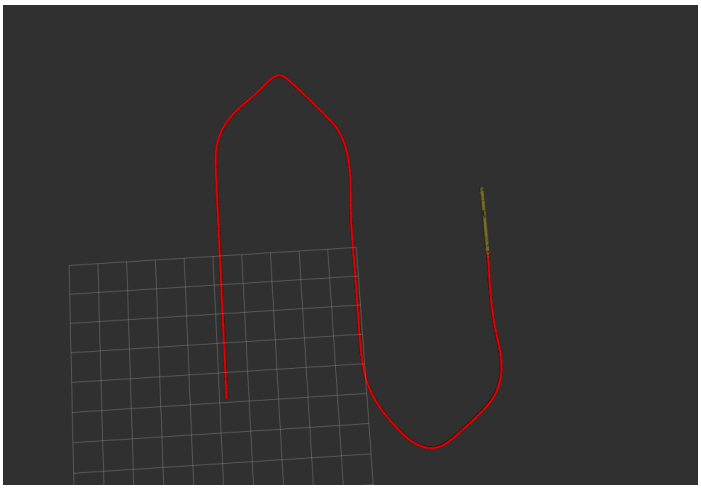


Figure 2.12: Waypoints describing a boustrophedon by adding a waypoint in each turn.

In the figure 2.11, the turn entry waypoint is never reached because the turn is too steep. In the second figure 2.12 a waypoint was added to each turn to make the turn smoother and force the AUV to run closer to the original waypoint.

By adding waypoints it is therefore possible to get closer to the desired trajectory for the main AUV while making it more feasible.

Controller

Thanks to the control interface, the main AUV has a "roadmap" to follow. To do this it is regulated in orientation, position, angular velocity and speed. This is a proportional and derivative controller. The proportional coefficient is related to the error in position and orientation and the derivative coefficient is related to the error in angular velocity and speed.

It should be noted that the regulation in orientation and angular velocity simply makes it easier to follow the trajectory in position. In other words, the priority in the regulation is convey in the tracking of the trajectory and not the direction of the trajectory. In the case of a local increase of the current, the AUV will be derived and will therefore be oriented towards the position it was supposed to be in. This point will be explained in the next section.

At the end of the mission, the controller goes into "hold position" mode. At any time during the mission, a new waypoint file can be loaded which erases the current path. This can be useful in case of an emergency or a change of search area.

Position in formation

With the main AUV's trajectory control assured, it must now send the positions to the other AUVs. To do this, it is necessary to know the formation's reference position, which correspond to the postion of the main AUV. The positions of the other AUVs depend on the direction of the path of the main AUV. It is necessary to differentiate this direction from the actual orientation of the main AUV. It can not be the one indicated by its AHRS or the one computed from GPS data.

Indeed if an error during regulation occurs or if the main AUV is derived, this will change the direction of the formation, which should not be the case. Because if the direction changes, the position of the AUVs in the formation will change which can impact the regulation of the other AUVs and thus the quality of the data.

The direction of the formation must be strictly the same as the direction of the trajectory of the main AUV. This trajectory being a succession of 'TrajectoryPoint' as explained previously, a direction vector can be known as the trajectory direction. The control interface allows us to know the position of a 'TrajectoryPoint' at a given moment. After catching this value for the present and the previous moment, we can determine this direction vector.

With this steering vector, we can place the other AUVs behind the main AUV as shown in the figure 2.13.

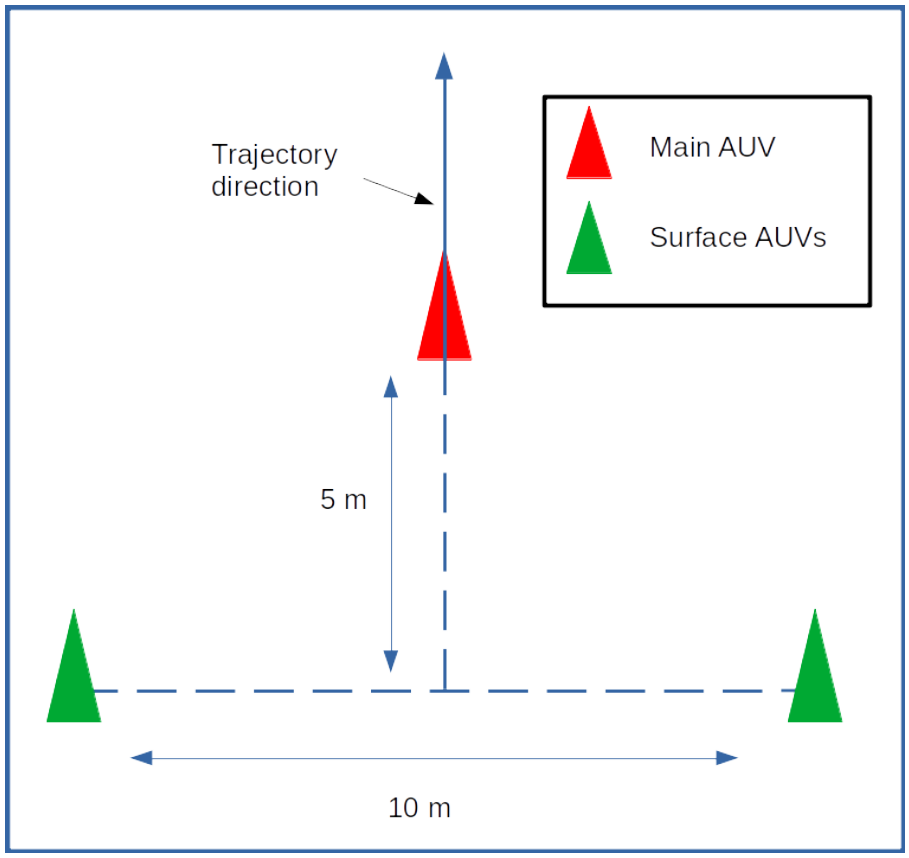


Figure 2.13: Position of the surface AUVs in formation

Only the position of the AUVs on the surface is shown to avoid overloading the diagram. The choice of distances depends on the physics of our AUV. In our case, the X300 has a total size of 3m so taking into account the inertia of the robot, which gives for our figure a spacing between the AUVs of about 7m. These distances are configurable and must be modified to avoid any collisions.

Some mathematical notions are to be specified to explain how to obtain the positions of the AUVs. We can position the AUVs 5 meters behind the main AUV thanks to the direction of the trajectory. On the other hand, to place the AUVs on the sides we need a direction vector orthogonal to the direction of the trajectory. The procedure to obtain it is explained above:

V is our steering vector of the trajectory and U is the orthogonal vector.

$$V = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}; U = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \tag{2.1}$$

The scalar product of U and V is therefore equal to 0 because they are orthogonal.

$$x_1 x_2 + y_1 y_2 = 0 \tag{2.2}$$

Which results in:

$$\begin{cases} y_2 = 0, x_2 = 1, \text{if } x_1 = 0 & (2.3) \\ x_2 = 0, y_2 = 1, \text{if } y_1 = 0 & (2.4) \\ x_2 = 1, y_2 = -x_1/y_1, \text{otherwise.} & (2.5) \end{cases}$$

With this orthogonal vector we can determine our two positions for the surface AUVs. On the other hand we have to determine which one is on the left of the main AUV and which one is on its right. To do this we will consider A and B the two positions obtained previously and M the position of the main AUV. Looking at the figure 2.14. We can see that if the determinant of (AM,BM) is positive then A is on the left and B is on the right. The reverse is true if the determinant of (AM,BM) is negative.

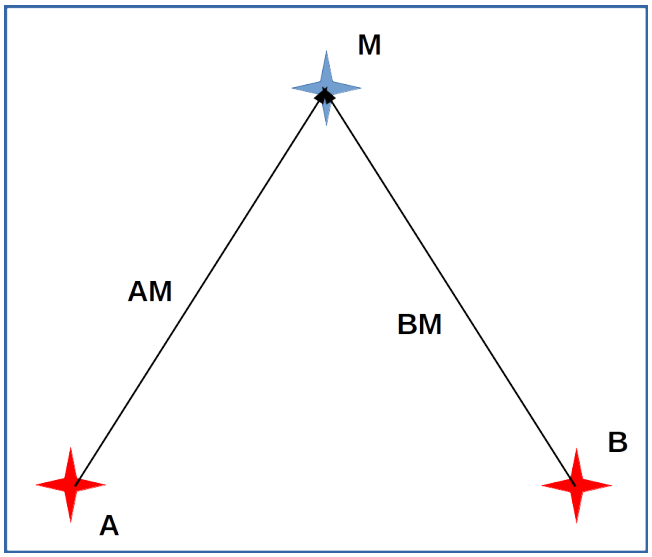


Figure 2.14: Determination of the position using determinant

Using these mathematical arguments, we ensure that the AUV swarm will be well oriented and that the positions of the surface AUVs will not be exchanged.

Thanks to the RVIZ[3] 3D viewer, you can display these positions to see the result. It can be observed that the formation is rather stable in straight lines. On the other hand, curves impose a sudden and discontinuous change of position.

[3]ROS Visualization

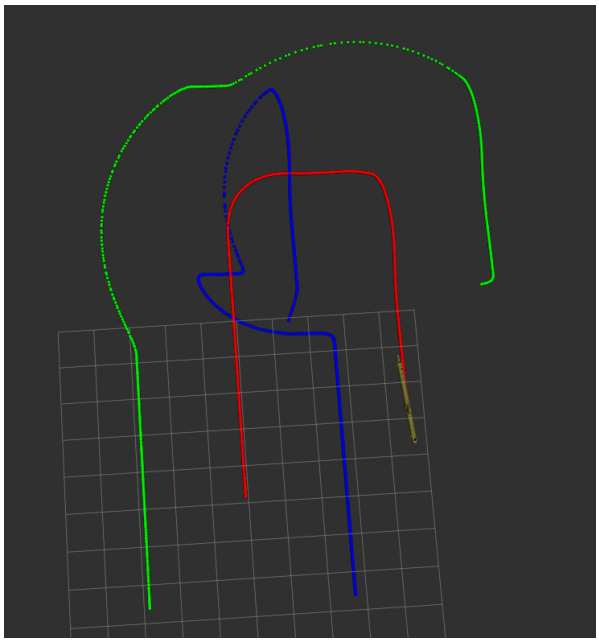


Figure 2.15: Position display on RVIZ without waypoints in the turn

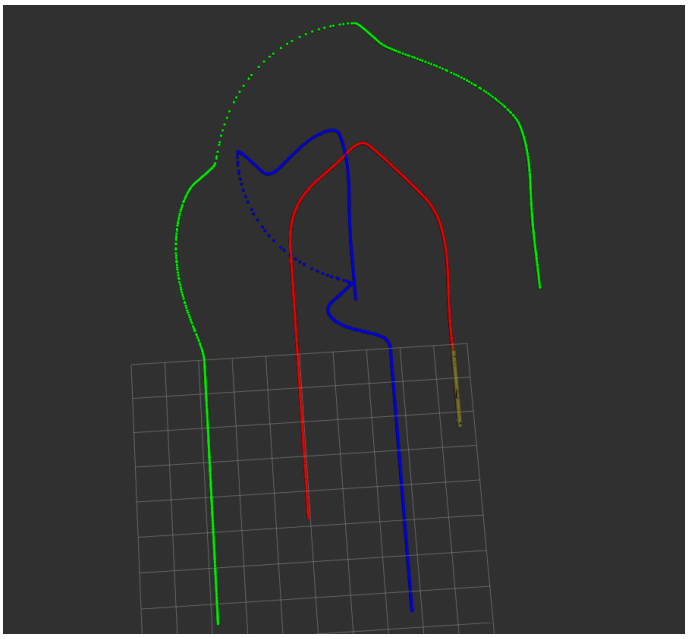


Figure 2.16: Position display on RVIZ with one waypoint in the turn

It can be seen from these figures that the more the main AUV undergoes large turns the more this will impact the position of the AUVs in the formation. We will see later how to improve this and how these new trajectories can be followed by the other AUVs.

2.2.2 Other AUVs

Unlike the main AUV, the AUVs are regulated to follow a setpoint position that corresponds to their position in the formation led by the main AUV. It is therefore not possible to operate the control interface as with the leading AUV. The controller can therefore only have a desired position as setpoint. We will therefore explain the operation of the controller, which we remind you that it cannot be regulated in roll.

Controller

For surface and diving AUVs, any setpoint is given in speed or angular velocity. It is also possible to calculate a setpoint orientation for our AUVs. Indeed if we know the setpoint position as well as the position of the AUV, we can calculate a setpoint direction vector for the AUV. This direction vector must then be converted into an orientation in order to be able to compare it with the current orientation of the AUV.

Convert 3D vector into euler angles

The attitude of an AUV is described by three angles, called Euler angles. These angles describe the rotations of the main axis of the AUV with respect to the original reference mark. A particularity is to be mentioned concerning this original marker. The NED[4] (North East Down) marker is used to have the positive vertical z axis towards the bottom while keeping an orthonormal marker.

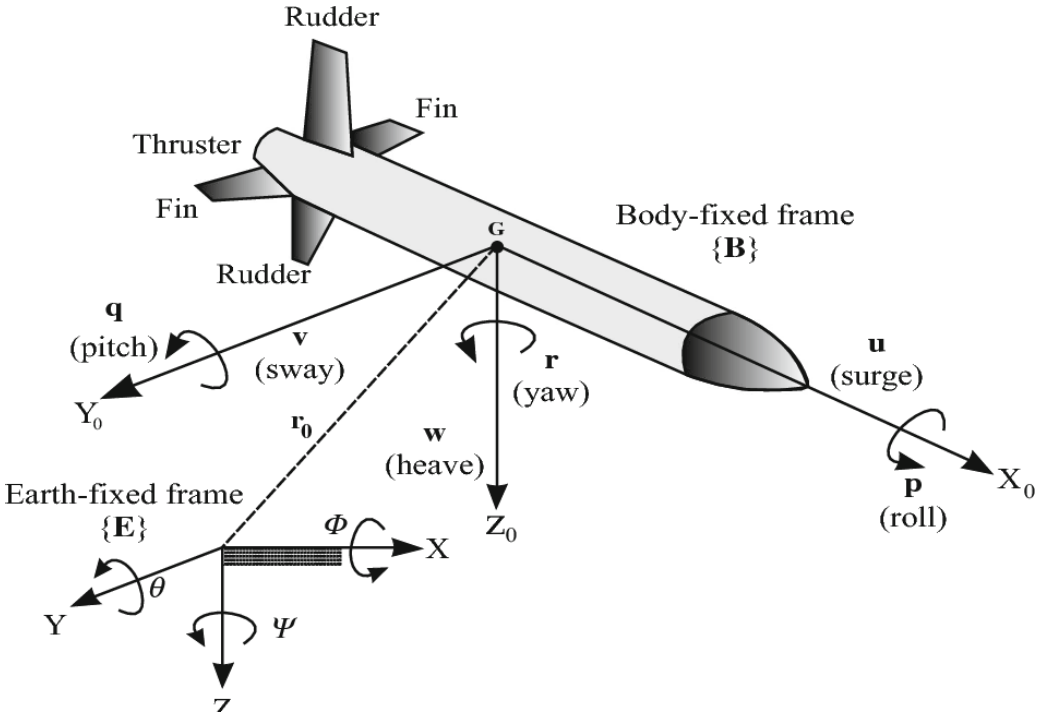


Figure 2.17: Euler's angle diagram for an AUV[5]

In our case, the AUV should not be rolling as this could impact the quality of the data acquired by the sonars. It is therefore considered that the target direction does not involve roll. At this point, we can see from figure 2.17 that yaw represents the desired heading and pitch will allow us to maintain the AUV at a constant depth.

The calculation of yaw and pitch is therefore done as follows: The steering vector V is defined in the Gazebo simulator marker. Which is similar to a NWU[5] (North West Up) oriented marker. So we change the vector marker:

$$V[\text{NWU}] = \begin{bmatrix} x \\ y \\ z \end{bmatrix} <=> V[\text{NED}] = \begin{bmatrix} x \\ -y \\ -z \end{bmatrix} \tag{2.6}$$

Considering V as the norm of the steering vector, we can project our vector into the NED

[4]North East Down
[5]North West Up

coordinate system thanks to the euler angles (roll $=\gamma$, pitch $=\alpha$, yaw $=\beta$):

$$V[\text{NED}] = \begin{bmatrix} Vcos(\alpha)cos(\beta) \\ Vcos(\alpha)sin(\beta) \\ -Vsin(\alpha) \end{bmatrix} \tag{2.7}$$

At this point we can see that no roll command can be obtained from our steering vector. This is also not a problem because we know that our AUV cannot be regulated in roll because no thruster is able to influence the roll. On the other hand we will see that it is important to keep the roll value at 0 for diving AUVs during the data acquisition phases. We will come back to this point later.

By identification, we get:

$$\begin{cases} Vcos(\alpha)cos(\beta) = x & (2.8) \\ Vcos(\alpha)sin(\beta) = -y & (2.9) \\ Vsin(\alpha) = z & (2.10) \end{cases}$$

This results in:

$$\begin{cases} \alpha = arcsin(z/V) & (2.11) \\ \beta = arctan(-y/x), \text{if } x \neq 0 & (2.12) \\ \beta = sgn(-y)\pi/2, \text{if } x = 0 & (2.13) \end{cases}$$

Pitch (α) will be given between -$\pi/2$ and $\pi/2$ as a result of the arcsin function. This result is perfect in our use because higher or lower angles would cause the AUV to turn over. Regarding the yaw angle (β), the pitch angle must be different from $\pm\pi/2$.

By using the arctan function, β angle is between -$\pi/2$ and $\pi/2$. Except that the yaw angle can go from -π to π. Indeed angle values greater than $\pi/2$ in absolute value can be obtained if x is negative.

The end result is:

$$\begin{cases} \alpha = arcsin(z/V) & (2.14) \\ \beta = arctan(-y/|x|), \text{if } x > 0 & (2.15) \\ \beta = \pi - arctan(-y/|x|), \text{if } x > 0 & (2.16) \\ \beta = sgn(-y)\pi/2, \text{if } x = 0 & (2.17) \end{cases}$$

Proportional and derivative controller

Our controller is therefore regulated thanks to a position setpoint and an orientation setpoint. It is not possible to regulate on the speed so the derivative parameter will not be used for the

error in speed or angular velocity. We have therefore chosen to make a cascade PD controller. The derivative coefficient will therefore be linked to the derivative of error.

Since the robots are controlled by force and torque setpoints, we have a control law of this type:

$$X[\text{NED}] = \begin{bmatrix} x \\ y \\ z \\ \alpha \\ \beta \\ \gamma \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix} ; \dot{X}[\text{NED}] = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \\ f_x \\ f_y \\ f_z \\ t_x \\ t_y \\ t_z \end{bmatrix} \tag{2.18}$$

$$\begin{cases} (f_x, f_y, f_z) \text{ are the forces in the local frame of the AUV} & (2.19) \\ (t_x, t_y, t_z) \text{ are the torques in the local frame of the AUV} & (2.20) \end{cases}$$

We must therefore determine the forces and torques resulting from a return of state through errors.

Considering P_t and R_t the target position and orientation, P_r and R_r the real position and orientation of the AUV. We have:

$$P_c[\text{NWU}] = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} ; R_c[\text{NWU}] = \begin{bmatrix} \alpha_c \\ \beta_c \\ \gamma_c \end{bmatrix} ; P_r[\text{NWU}] = \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} ; R_r[\text{NWU}] = \begin{bmatrix} \alpha_r \\ \beta_r \\ \gamma_r \end{bmatrix} \tag{2.21}$$

This results in the following errors:

$$E_{position}(t) = \begin{bmatrix} Ep_x(t) \\ Ep_y(t) \\ Ep_z(t) \end{bmatrix} = \begin{bmatrix} x_c(t) - x_r(t) \\ y_c(t) - y_r(t) \\ z_c(t) - z_r(t) \end{bmatrix} ; E_{orientation}(t) = \begin{bmatrix} Eo_\alpha(t) \\ Eo_\beta(t) \\ Eo_\gamma(t) \end{bmatrix} = \begin{bmatrix} \alpha_c(t) - \alpha_r(t) \\ \beta_c(t) - \beta_r(t) \\ \gamma_c(t) - \gamma_r(t) \end{bmatrix} \tag{2.22}$$

This gives the following derivative errors, with $\triangle t$ the time between two measurements. In our case, the time between two error measurements is related to the frequency of publication of formation positions. This frequency is equal to 120Hz on average which gives a $\triangle t$ of less

than 0.01 seconds. In our case this is sufficient, this $\triangle t$ can be increased in case we notice that the AUV performs too many calculations.

$$Ed_{position} = \begin{bmatrix} Edp_x(t) \\ Edp_y(t) \\ Edp_z(t) \end{bmatrix} = \begin{bmatrix} \frac{Ep_x(t) - Eo_x(t-\triangle t)}{\triangle t} \\ \frac{Ep_y(t) - Eo_y(t-\triangle t)}{\triangle t} \\ \frac{Ep_z(t) - Eo_z(t-\triangle t)}{\triangle t} \end{bmatrix} \tag{2.23}$$

$$Ed_{orientation} = \begin{bmatrix} Edo_\alpha(t) \\ Edo_\beta(t) \\ Edo_\gamma(t) \end{bmatrix} = \begin{bmatrix} \frac{Eo_\alpha(t) - Eo_\alpha(t-\triangle t)}{\triangle t} \\ \frac{Eo_\beta(t) - Eo_\beta(t-\triangle t)}{\triangle t} \\ \frac{Eo_\gamma(t) - Eo_\gamma(t-\triangle t)}{\triangle t} \end{bmatrix} \tag{2.24}$$

This therefore requires the value of the previous error to be recorded. All these errors are calculated in the NWU marker. These errors should not be calculated in the local robot coordinate system, because the position error values would not be comparable. At this point we can define the set forces as:

$$F[\text{NWU}](t) = \begin{bmatrix} F_x(t) \\ F_y(t) \\ F_z(t) \end{bmatrix} = \begin{bmatrix} Kp_x Ep_x(t) + Kd_x Edp_x(t) \\ Kp_y Ep_y(t) + Kd_y Edp_y(t) \\ Kp_z Ep_z(t) + Kd_z Edp_z(t) \end{bmatrix} \tag{2.25}$$

$$T[\text{NWU}](t) = \begin{bmatrix} T_\alpha(t) \\ T_\beta(t) \\ T_\gamma(t) \end{bmatrix} = \begin{bmatrix} Kp_\alpha Eo_\alpha(t) + Kd_\alpha Edo_\alpha(t) \\ Kp_\beta Eo_\beta(t) + Kd_\beta Edo_\beta(t) \\ Kp_\gamma Eo_\gamma(t) + Kd_\gamma Edo_\gamma(t) \end{bmatrix} \tag{2.26}$$

In addition, the thruster manager allows the AUV to be controlled via forces which are defined in the local reference frame of the relevant AUV. We must therefore apply a rotation to this force vector to obtain the desired force vector in the AUV reference frame. The rotation applied is the rotation of the euler angles we know because it is the orientation of the robot at the same instant.

$$R[\alpha, \beta, \gamma](t) = \tag{2.27}$$

$$\begin{bmatrix} cos(\gamma) * cos(\beta) & cos(\gamma) * sin(\beta) * sin(\alpha) - sin(\gamma) * cos(\alpha) & cos(\gamma) * sin(\beta) * cos(\alpha) + sin(\gamma) * sin(\alpha) \\ sin(\gamma) * cos(\beta) & sin(\gamma) * sin(\beta) * sin(\alpha) + cos(\gamma) * cos(\alpha) & sin(\gamma) * sin(\beta) * cos(\alpha) - cos(\gamma) * sin(\alpha) \\ -sin(\beta) & cos(\beta) * sin(\alpha) & cos(\beta) * cos(\alpha) \end{bmatrix} \tag{2.28}$$

This results in:

$$F[\text{AUV}frame](t) = \begin{bmatrix} f_x(t) \\ f_y(t) \\ f_z(t) \end{bmatrix} = R[\alpha, \beta, \gamma](t) F[NWU](t) \tag{2.29}$$

We will then see in the last part, dealing with the results obtained, which values have been retained for the parameters of our controller.

Chapter 3

Results and improvements

In this section we will present the different results obtained during the developpment of this project. We will also explain what are the tracks of evolution or improvement planned.

3.1 Controller

The controller was central during this internship and allowed, as it evolved, to find out ways to improve certain parts of the project.

3.1.1 Parameters

In order to choose the best proportional and derivative parameters, certain criteria for the control of our AUV must be specified.

It is obvious that the AUV must be reactive in order to compensate possible disturbances. If disturbances appear or the AUV enters a bend, we know that the AUV will automatically be distanced from the main AUV. The role of the proportional coefficient is therefore to allow the AUV a reactivity to catch back the target position. On the other hand, it is also necessary to avoid overshooting of the desired position. Because this would lead to an orientation error of about 180°. This would force the controller to give very important moment commands and would cause the AUV to drift. This is illustrated in the figure 3.1.

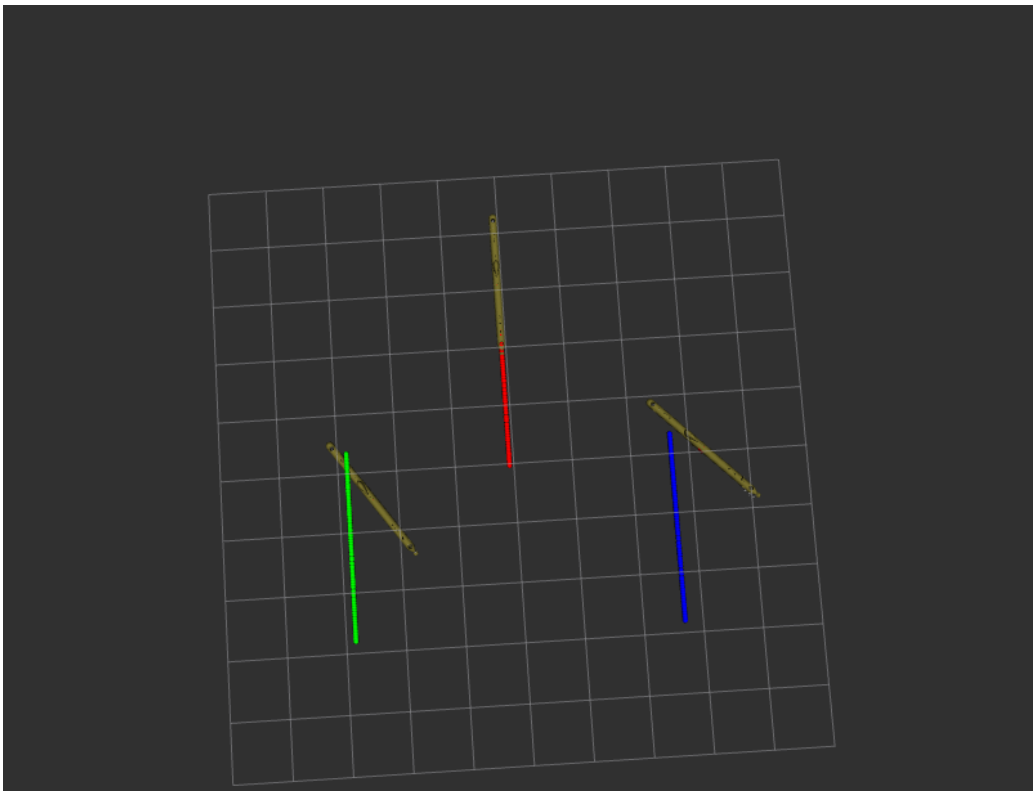


Figure 3.1: Overshoot in the position by the right AUV

The derivative coefficient will allow us to reduce these overshoots and also allows us to anticipate the inertia of our AUV. Unlike a controller with a fixed setpoint (e.g. position), our system has a position setpoint that changes at every moment. This allows us to have a margin on the coefficients because if we have an overshoot on a setpoint, it is very likely that the setpoint will have changed when the overshoot occurs.

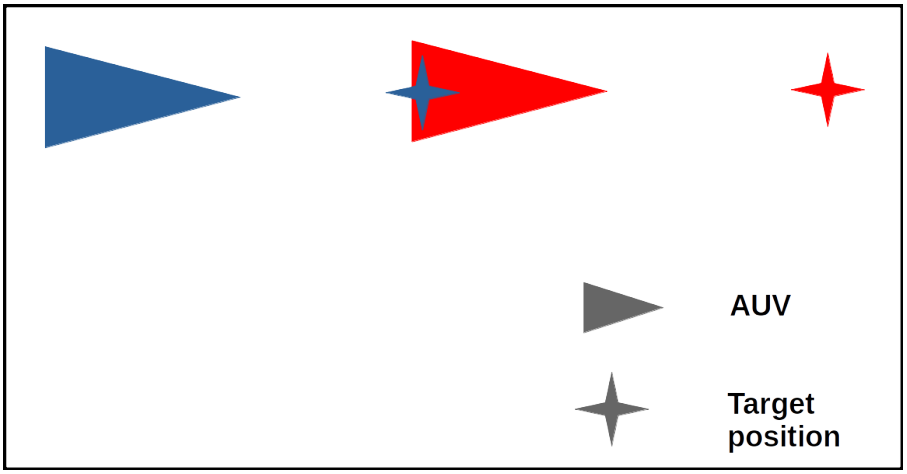


Figure 3.2: Target position and AUV at two different time

In the figure 3.2, we have our AUV following the desired position and at the next moment it has exceeded it but the setpoint position has moved over. A gain margin can therefore be

considered for the proportional coefficient which will make the system even more reactive.

In contrast to position control, orientation control must be carried out without overshooting. This is because the setpoint orientation changes very little in a straight line. Moreover, the straight line corresponds to the acquisition phase and the AUV must have a constant orientation during this phase. The controller must still be reactive so that it cannot continue to follow the setpoint during turns.

An important point is to be mentioned when searching for controller parameters. As previously presented, the AUV can apply forces along the axis perpendicular and horizontal to its main axis (Yo in Figure 2.17). But when one applies a force along this axis, the AUV kiosk will create friction which will cause a roll. This is to be avoided for diving AUVs. It is thus necessary to reduce as much as possible the regulation along the y-axis. To do this, either we reduce the regulation parameters or we start by finely regulating the orientation to force the AUV to mainly move in its main axis, the x axis.

From these criteria, and starting with the orientation regulation, we have obtained the following parameters:

$$Kp = \begin{bmatrix} Kp_x \\ Kp_y \\ Kp_z \\ Kp_\alpha \\ Kp_\beta \\ Kp_\gamma \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 0 \\ 4 \\ 4 \end{bmatrix} ; Kd = \begin{bmatrix} Kd_x \\ Kd_y \\ Kd_z \\ Kd_\alpha \\ Kd_\beta \\ Kd_\gamma \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 0 \\ 100 \\ 100 \end{bmatrix} \tag{3.1}$$

Controller in straight lines

The performance of the controller in a straight line can be seen in the following figures:

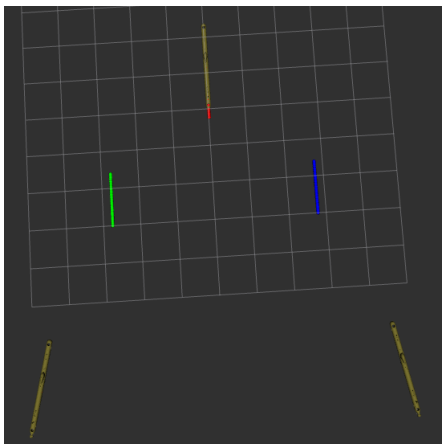


Figure 3.3: Left and Right AUVs controlled in straight line t_1

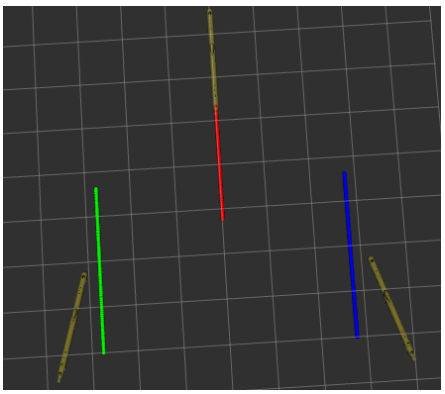


Figure 3.4: Left and Right AUVs controlled in straight line t_2

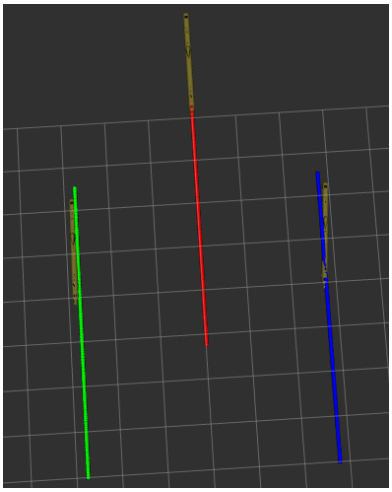


Figure 3.5: Left and Right AUVs controlled in straight line t_3

As can be seen, the orientation remains stable as well as the position control. There is no overshoot and the AUV is catching up by positioning itself just behind the set position. Proportional coefficients therefore allow the AUVs to catch up with a setpoint position. The derived coefficients for the position are low but allow a small overshoot for the AUV. On the other hand, as can be seen, the derivative coefficients for the orientation are very high and provide a very strong reduction of the overshoot as can be seen in the previous figures.

Controller in turn

The performance of the controller in turns can be evaluated with the figures in the appendix 3.2.2.

At the beginning of the turn, in the first figure, the AUV is at the desired position. As we could see in the figure 2.15, the turn of the main AUV will move the target position very quickly in a very short time. We can see that it only takes two moments for the two AUVs to be very far from the target position. The last figure in the appendix 3.2.2 shows that, at the end of the turn of the main AUV, the left AUV is far from its set point and the right AUV will end up in the opposite direction of the desired trajectory. For the right AUV, it will have to turn 180° which will delay the robot positioning.

3.1.2 Waypoint file

If we focus on the regulation of the diving AUVs, it is very important for them to be precisely at the requested position and in the right direction at the beginning of each line. However, we can see that during turns, the AUVs have difficulties to be positioned and oriented before starting the straight line.

In the previous section we mentioned the possibility to choose the speed of passing a waypoint. This speed can be set in the waypoint file. In order to avoid the AUVs being dropped when turning, this speed can be decreased to give the AUVs time to get closer to the desired position. The figures 3.6 and 3.7 shows that the setpoint positions are closer together, giving the AUVs more time to get closer to them.

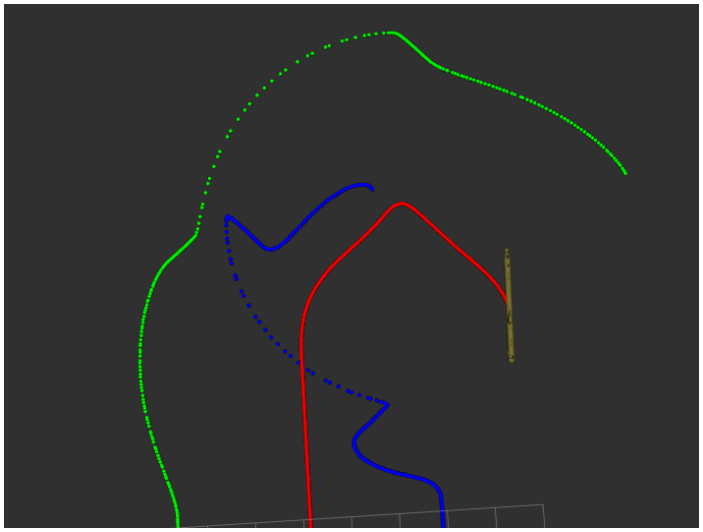


Figure 3.6: Formation in turn with speed at 0.4

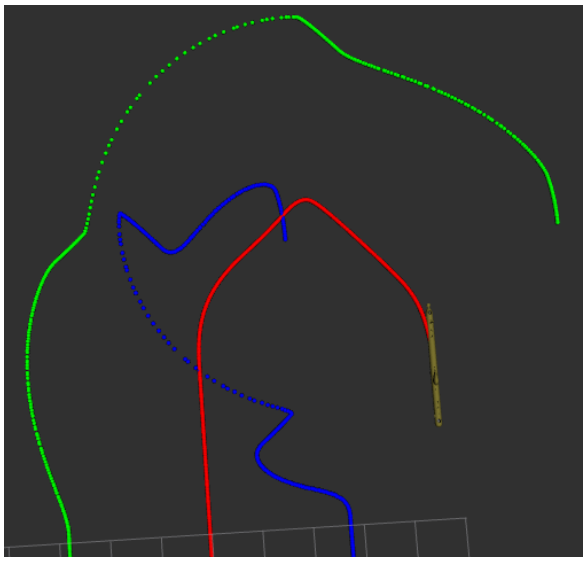


Figure 3.7: Formation in turn with speed at 0.05

The result is not significant even if the speed is reduced by a factor of 8. We can see that the main problem with these trajectories comes from the angles of turn which are too high for the main AUV. We can therefore modify the waypoints file by adding waypoints in the turn to decrease the turning angles.

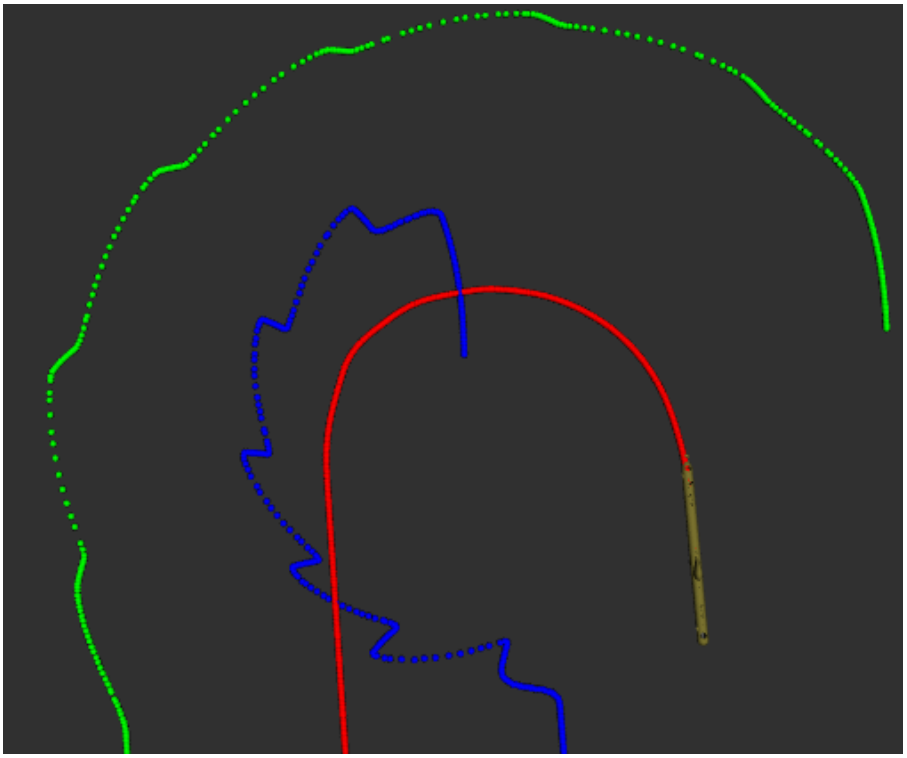


Figure 3.8: Turn with 5 waypoints added with speed at 0.4

In the figure 3.8, we can see that the trajectory of the main AUV is much smoother. This

also makes the trajectories of the other AUVs slightly smoother. The large trajectory jumps for the other AUVs have been split into several smaller jumps. On closer inspection, it can also be seen that the green and blue trajectories have moved away from each other, thus reducing the risk of collision between the AUVs. The third figure in the appendix 3.2.2 shows the left AUV crossing the path of the right AUV.

We can see that the more points we add, the smoother the AUVs' trajectory will be. It will be interesting to automatically add waypoints in turns to avoid this phenomenon of oscillations on the trajectory. Looking at the difference in trajectory between 1 and 5 added waypoints, we can assume that about ten waypoints would be enough to make the trajectory smooth.

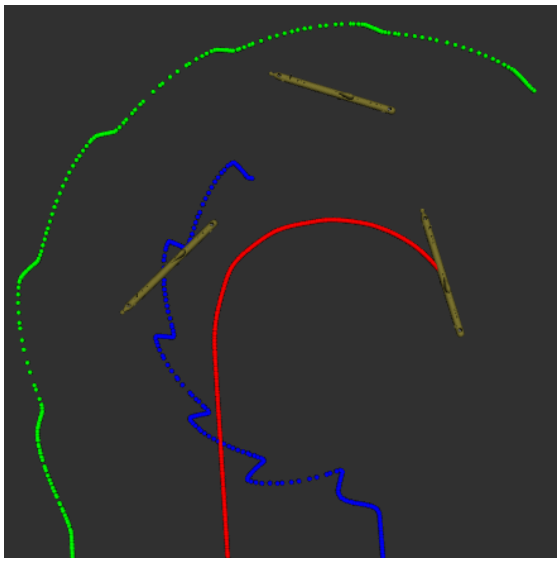


Figure 3.9: AUV's controller in turn with 5 waypoints at t_1

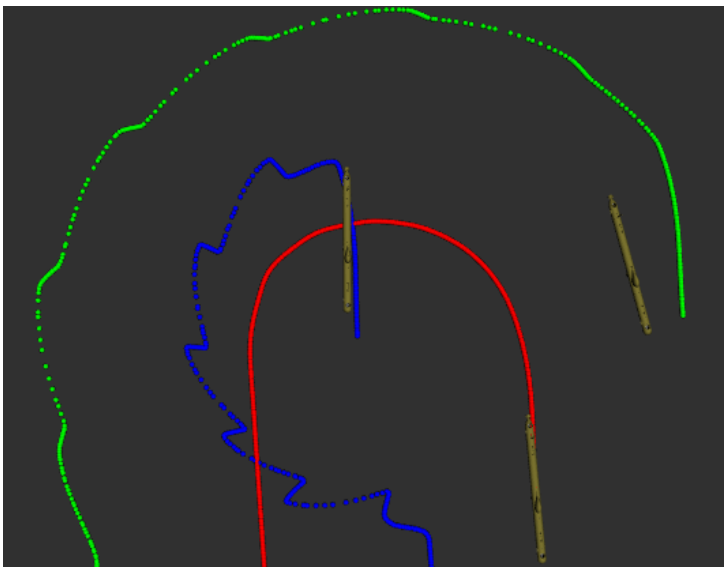


Figure 3.10: AUV's controller in turn with 5 waypoints at t_2

In the previous figures we can see that the smoothing of the trajectories during the turns allows the AUVs not to fall behind and to be well positioned at the beginning of the next line.

3.2 Improvements

Working conditions (tele-work, jet lag) during the COVID-19 period made it difficult to develop the project, which was just getting off the ground. The following are therefore possible improvements to the existing algorithms and improvements to be made to the project.

3.2.1 Controller

In our project, we chose not to add an integral coefficient. Because of this permanent change of setpoint, the integral part will always be of the same sign. This would therefore lead to unwanted overshoots. Therefore, the PD coefficients should only be improved but no integral parameters should be added.

As we said before, an improvement of the waypoint files during the mission preparation will facilitate the regulation of the AUVs. This improvement will greatly help for the quality of the data.

Furthermore, we have observed that the AUV controller allows to follow changes in angle and position in relatively short time. Looking at the thruster characteristics of the X300 AUV, it seems very unlikely that an AUV such as this one would be able to perform this type of control.

In order to make our system more realistic, we have added a saturation coefficient corresponding to the characteristics of the engines. This parameter did not bring any modification to the regulation of our AUVs. Furthermore, we must take into account that the forces applied to the thrusters are transmitted instantaneously to the vehicles because we are in a simulation. Which is not the actual behavior of a robot as we can see in the figure 3.11

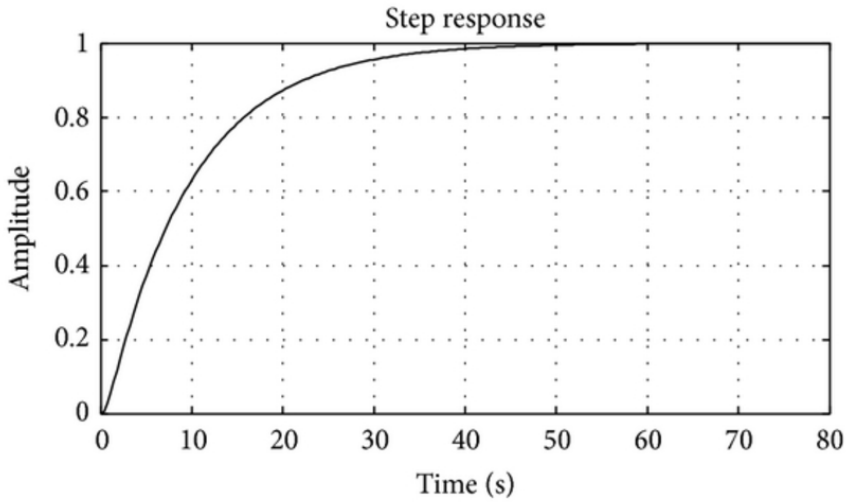


Figure 3.11: Typical step response of a DC motor

However, it is obvious that the force setpoints given to the thrusters correspond to a rotation speed of the thrusters which is not which is not instantly transmitted..

To make our system closer to the actual and real model, it would be a good idea to add a delay on the application of the thruster controls. This would allow us to better account for the behaviour of the thrusters. This delay can be estimated by real tests or by studying the step response of the thruster motors.

3.2.2 Kalman filter

As we have just mentioned, the controller has performances in terms of stability, speed and precision. These performances are mainly due to the accuracy of the odometric data supplied to the AUV. Indeed, in the development of our controller, we used odometric data from the Gazebo simulator. Indeed, these values are accurate and are provided with a high frequency.

However, it is certain that these odometry values cannot be as accurate during a mission at sea. They are generally the result of a kalman filter that must be developed on the basis of information from the IMU and GPS (and other sensors if possible).

As we discussed in the first part on the state of the art, the MONSUN Project explains in its work[7] the importance of kalman filters for localization[1]. Acoustic communication modems[8] are essential for the trilateration of AUVs underwater.

We have chosen at the beginning of the project to surround the underwater AUVs with three surface AUVs but it is obvious that it is preferable to use only two. This would reduce the risk of collision not between surface and underwater robots because they do not move at the same depth. But it would decrease the risk of collision between the AUVs that we have shown in the figure 3 in the appendix 3.2.2. This can only be done after implementing a precise kalman filter based on information from the IMU, the positions of the other AUVs and the sonar measurements.

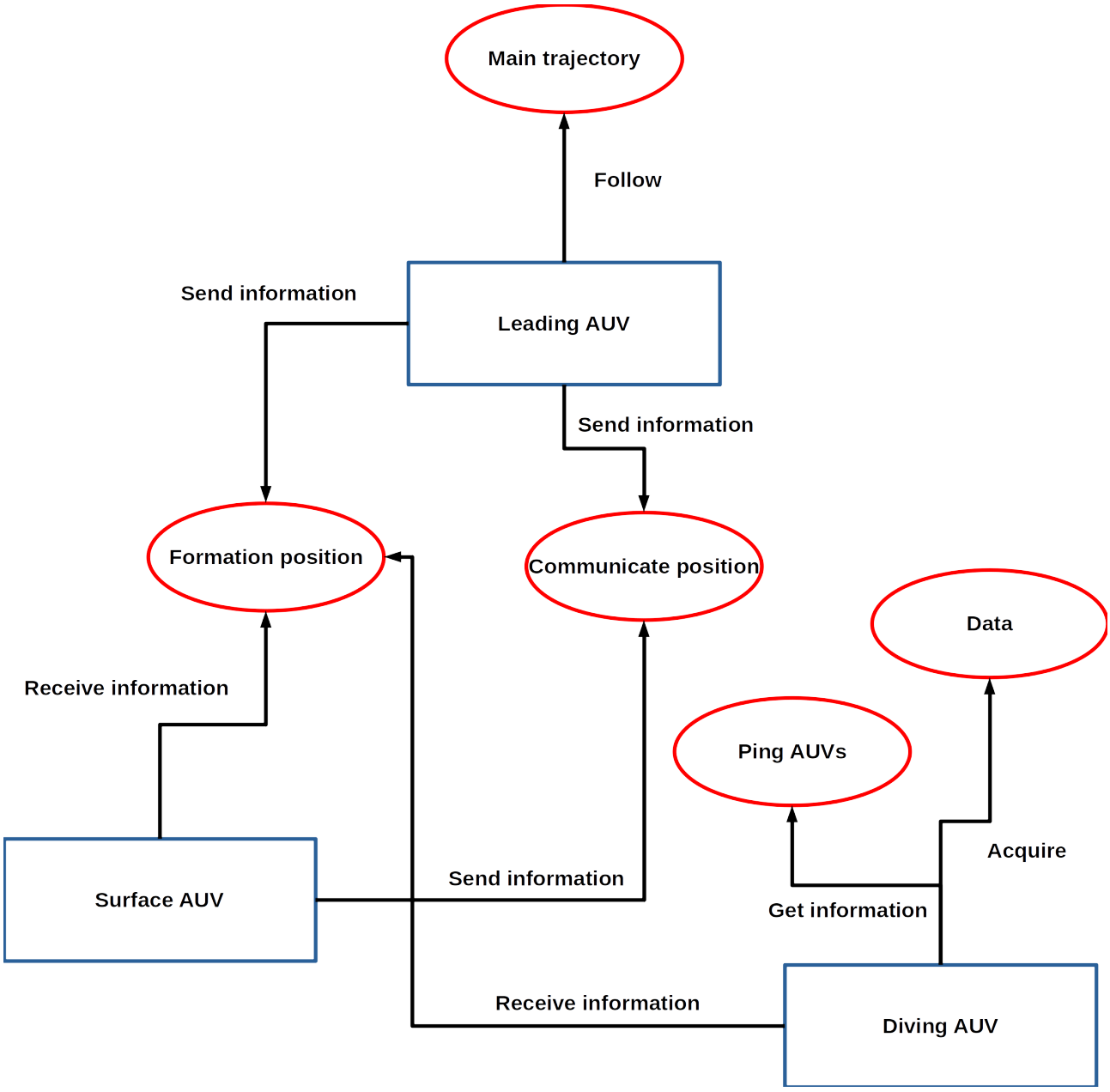
Conclusion

This internship is part of a global project of autonomous robotics concerning all aspects of this field. I was able to choose a more specific part to start with, which allowed me in particular to take advantage of the lessons in robot simulation that I had at ENSTA Bretagne. Nevertheless, I had to take into account all aspects of the project throughout the development of the control algorithms.

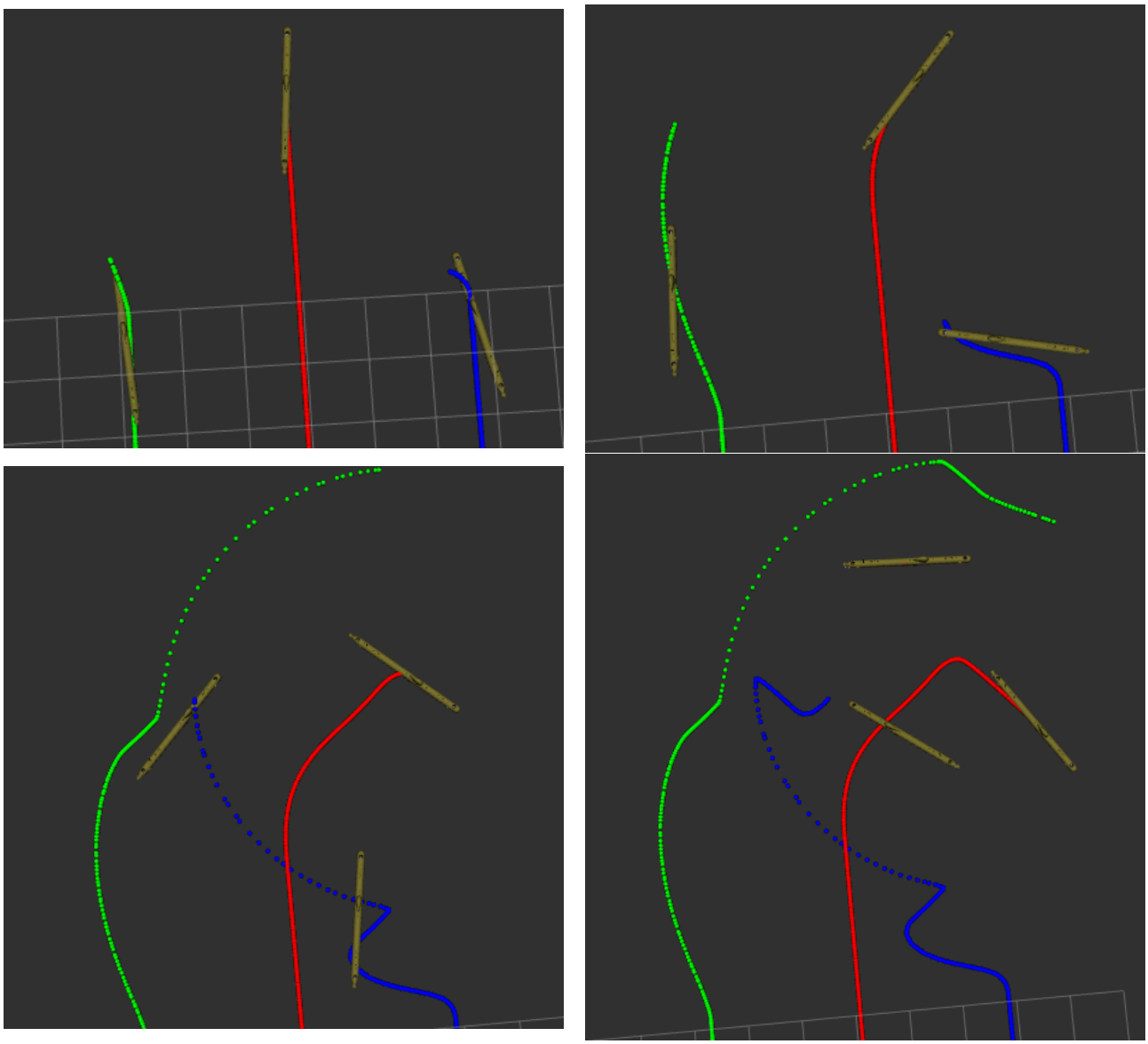
My work in this project allowed me to understand the issues related to the regulation of a system evolving among a swarm of robots. In an English-speaking context, I was also able to train my skills in terms of expression, precisely in a remote work context. This working context has forced me to reorganize and review the objectives of the internship in order to be the most efficient.

The Gazebo simulation software coupled with the ROS middleware allowed me to develop a controller that meets the performance criteria required for the X300 AUV. Namely stable, fast, precise and collision avoidance. The resulting simulations highlighted possible improvements to this control system.

Functional architecture of the project



AUV controller in turn



Glossary

AUV Autonomous Underwater Vehicle

GNSS Global Navigation Satellite System

GPS Global Positioning System

IMU Inertial Measurement Unit

LIPD Linear Interpolator with polynomial blend

NED North East Down

NWU North West Up

ROS Robot Operating System

RVIZ ROS Visualization

Bibliography

[1] B. Allotta, R. Costanzi, E. Meli, L. Pugi, A. Ridolfi, and G. Vettori. Cooperative localization of a team of auvs by a tetrahedral configuration. *Robotics and Autonomous Systems*, 62(8):1228–1237, 2014.

[2] Ammar Amory, Benjamin Meyer, Christoph Osterloh, Thomas Tosik, and Erik Maehle. Towards fault-tolerant and energy-efficient swarms of underwater robots. *2013 IEEE International Symposium on Parallel and Distributed Processing, Workshops and Phd Forum*, 2013.

[3] Ulrich Behrje, Cedric Isokeit, Benjamin Meyer, and Erik Maehle. A robust acoustic-based communication principle for the navigation of an underwater robot swarm. *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO)*, 2018.

[4] Tim Kridel. The steep learning curve for drone light shows. *Inavate*, 2019.

[5] P.s. Londhe, B.m. Patre, L.m. Waghmare, and M. Santhakumar. Robust proportional derivative (pd)-like fuzzy control designs for diving and steering planes control of an autonomous underwater vehicle. *Journal of Intelligent and Fuzzy Systems*, 32(3):2509–2522, 2017.

[6] Musa Morena Marcusso Manhães, Sebastian A. Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach. UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*. IEEE, sep 2016.

[7] Benjamin Meyer, Christian Renner, and Erik Maehle. Versatile sensor and communication expansion set for the autonomous underwater vehicle monsun. *Advances in Cooperative Robotics*, page 250–257, 2016.

[8] Christian Renner, Alexander Gabrecht, Benjamin Meyer, Christoph Osterloh, and Erik Maehle. Low-power low-cost acoustic underwater modem. *Quantitative Monitoring of the Underwater Environment Ocean Engineering and Oceanography*, page 59–65, 2016.

[9] Peter Simon. Military robotics: Latest trends and spatial grasp solutions. *International Journal of Advanced Research in Artificial Intelligence*, 4(4), 2015.