An Automatic Docking System for AUVs with applications for the inspection of offshore aquaculture and floating wind energy infrastructures

Guilherme Jorge de Menezes Crespo Ferreira

.

Instituto Superior Técnico, Lisboa, Portugal

June, 2024

***Abstract*—This thesis focuses on the development of an autonomous docking procedure for Autonomous Underwater Vehicles (AUVs). The need for such a procedure arises from the increasing demand for efficient and autonomous inspection of offshore aquaculture and floating wind energy infrastructures. Due to the expected increase in the world's population, the demand for renewable energy and sustainable food resources is at an all-time high. Docking systems play a pivotal role in enhancing AUV autonomy, providing safety in adverse conditions, recharging, data transmission, and prompt deployment during long continuous operations.**

This research recognizes the complexity of the underwater environment, which poses unique challenges to traditional communication and localization methods. Conventional methods, such as communication via electromagnetic waves, are limited by the depth and unpredictability of the ocean. Thus, this study delves into the analysis of suitable underwater localization methods, ensuring that the AUV can navigate accurately in and out of the docking station. Among these methods, such as sonar and acoustic communication, image-based visual servoing stands out as the preferred option for guiding the vehicle due to its simplicity in terms of infrastructure and installation. The integration of the advanced Yolov8 convolutional neural network from Ultralytics allows for the analysis of images from the onboard camera of the AUV, extracting crucial features of the docking structure used in external control. This control strategy generates reference velocities based on image features, providing an intuitive and powerful approach, as long as the environmental visibility is satisfactory.

***Index Terms*—Autonomous Docking, Underwater Vehicle Control, Image-based control, Object and Pose Detection, Convolutional Neural Network.**

I. Introduction

On our planet, over 70% of the Earth's surface is covered in water. Considering that humanity has been harvesting nature's resources from less than 30% of the Earth's surface for two hundred thousand years, one can't help but wonder what large deposits of natural resources might exist at the ocean's bottom. Even without considering these resources, the study and survey of this extensive body of water influence our understanding of global and local climates, as well as other economic and social phenomena impacted and reflected upon by the marine ecosystem. It is fair to say that this vast portion of our planet remains underutilized to this day. With the expected increase in the world's population, demands for renewable energy and food resources are at an all-time high. With that in mind, the ocean's vastness is poised to see a surge in offshore aquaculture and floating wind energy infrastructures [1] in order to more efficiently optimize our use of space on Earth. There is a pressing need to eliminate redundant, repetitive, and at times perilous tasks performed by humans and, instead, automate these tasks while simultaneously improving efficiency and safety, as traditionally provided by human operators. The aforementioned offshore aquaculture and floating wind energy infrastructures serve as examples of this. They necessitate inspection and extended, continuous monitoring, often in remote locations with limited human presence.

Autonomous Underwater Vehicles (AUVs) offer promising solutions, reducing costs, risks, and human presence in hazardous environments. While Remotely Operated Vehicles (ROVs) provide safety, their tethering limits maneuverability. Autonomous docking systems represent a crucial advancement, enabling extended missions and data transmission with the surface.

Yet, underwater environments pose unique challenges for navigation and communication, requiring innovative solutions for successful docking procedures. Thus, this work aims to develop and test systems for docking AUVs onto stations, focusing on algorithms for safe navigation using visual and acoustic cues, with simulations as a primary testing ground.

A. Problem Description and Objectives

The primary objective of this work is to study, develop, and simulate a set of control tools that enable an AUV to locate, navigate, and safely dock/undock from an underwater docking station. In the literature, this problem is commonly divided into two distinct phases: the 'docking' procedure and the 'homing' phase. The homing phase is essentially aimed at positioning the AUV close enough to initiate the docking procedure. As a result, the work to be carried out will be segmented into multiple sub-problems:

- **Vehicle Modeling and Controller Design** - This section will encompass the description of the AUV's motion within its respective environment using a series of kinematic and dynamic equations. It will also define the controllers employed to track the relevant reference signals.

- **Homing Stage** - This phase's goal is to ensure that the AUV can autonomously locate the docking station within a specified range and align itself successfully with it before proceeding to the next phase of the procedure.
- **Docking (and deployment) Stage** - In this phase, the objective is to guarantee that the AUV can autonomously and safely steer itself into (and out of) the dock.

II. System Modeling

To obtain the vehicle model, it is necessary to define the reference frames and the notation used throughout this document. First, the Inertial Reference Frame, {U} is composed by the axes $\{\ x_U, y_U, z_U\ \}$. This reference is frame is used for measurements and movements relative the the world's terrain. That means it is essentially "attached" to any fixed place on Earth, assuming that the accelerations of a point at the surface of the Earth can be neglected. The axis follows the North-East-Down (NED) reference convention, meaning that x_U points north, y_U points East and z_U points downwards. Second, the Body Reference Frame, {B} is composed by the axes $\{\ x_B, y_B, z_B\ \}$. This reference frame is attached to the vehicle's center of mass, and its axis correspond to the vehicle's principal axis of inertia. The Figure 1 showcases all of these reference frames.

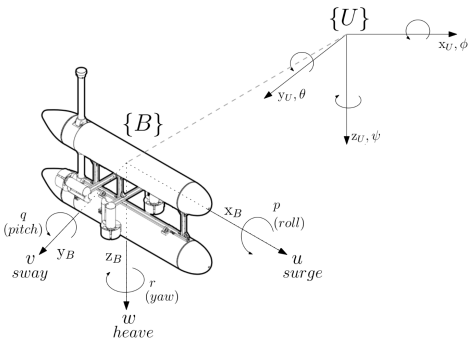

Fig. 1: Adopted reference frames, from [2].

For ease of study between multiple works, the nomenclature defined by the SNAME for treating motion of a submerged body through a fluid will be used. Further more, for the work developed, the AUV does not require the usage of all 6 degrees of freedom (DOF) of the 3D world. That is because of the design of the Dock used, further explored in section V and because of the nature of the vehicle. The dock and its dimensions were designed to encapsulate the vehicle with its roll and pitch equal to zero. Since the vehicle is naturally stable on these axis it is unnecessary in this work for the AUV to have a pitch or roll other than $0°$ to successfully complete its mission. Since these values, $\theta = \phi = 0°$, the total degrees of freedom of the vehicle's movement is decreased to 4: $\eta = [x, y, z, \psi]^T$. As such, the vehicle simplified kinematic model is given by

$$\begin{aligned} \dot{x} &= u\cos(\psi) - v\sin(\psi) \\ \dot{y} &= u\sin(\psi) + v\cos(\psi) \\ \dot{z} &= w \\ \dot{\psi} &= r. \end{aligned} \tag{1}$$

The dynamics of the system are also simplified by neglecting roll and pitch and their torques, resulting in

$$\begin{cases} m_u\dot{u} - m_v vr + d_u u = \tau_u \\ m_v\dot{v} + m_u ur + d_v v = \tau_v \\ m_w\dot{w} + d_w w - mg + B_f = \tau_w \\ m_r\dot{r} - m_{uv}uv + d_r r = \tau_r \end{cases}, \tag{2}$$

where

$$\begin{aligned} m_u &= m - X_{\dot{u}}, & d_u &= -X_u - X_{u|u|}|u|, \\ m_v &= m - Y_{\dot{v}}, & d_v &= -Y_v - Y_{v|v|}|v|, \\ m_w &= m - Z_{\dot{w}}, & d_w &= -Z_w - Z_{w|w|}|w|, \\ m_r &= I_z - N_{\dot{r}}, & d_r &= -N_r - N_{r|r|}|r|, \\ m_{uv} &= m_u - m_v, & B &= V_{sub}\ \rho_{water}\ g. \end{aligned} \tag{3}$$

with m, m_u, m_v, m_w representing the mass and hydrodynamic added mass, d_u, d_v, d_w and d_r representing the hydrodynamic damping effects, B_f being the buoyancy force, V_{sub} the volume of displaced water by the submerged vehicle and ρ_{water} the density of water.

III. Motion Control

In order to control both the dynamic and kinematic model previously described, an outer-inner loop architecture was employed. At the inner loop level, the controllers operate on the Dynamic model. The reference signals received are the necessary to complete its mission (speeds, heading, pitch, and others) and the outputs are the calculated forces and torques required to actuate on the vehicles dynamics so that it follows the given references. These controllers are mostly PID controllers (Proportional, Integral, and Derivative). At the outer loop level, the controllers analyze the current kinematic state and the goal of the mission inputted and thus generate the correct reference signals to be tracked by the inner loop controllers. The design of the outer loop controller strongly depends on the task at hand and changes based on which mission the agent is expected to perform, while the inner controller can be generic for a wide range of missions. A block diagram of this strategy can be observed in Figure 2.

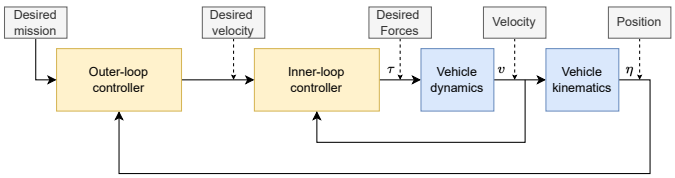

Fig. 2: Inner-Outer Loop Control design, adapted from [3]

At the inner-loop, the controllers for the surge, sway, heave and yaw-rate controller velocities are all PID, each with their

gains respectively tuned. All of their outputs is fed into the thrusters of the AUV, resulting in the desired forces and torques that follow the reference velocities.

IV. Image-Based Visual Servoing

A. Basics

Let $P_c = [x_c, y_c, z_c]^T$ be a 3D point in space expressed by the camera's frame. Its 2D projection on the image plane $p = [x_{img}, y_{img}]^T$ can be described by the pin-hole camera equation:

$$\begin{cases} x_{img} = \dfrac{x_c}{z_c} = (u - u_0)/p_x \\ y_{img} = \dfrac{y_c}{z_c} = (v - v_0)/p_y \end{cases} \tag{4}$$

where $\boldsymbol{m} = (u, v)$ gives the coordinates of the image point expressed in pixel units, and $\boldsymbol{a} = (u_0, v_0, p_x, p_y)$ is the set of camera intrinsic parameters, defined as:

- u_0 and v_0 - coordinates of the principal point, in pixels;
- p_x and p_y - ratio between the focal length and the size of a pixel.

Considering $\boldsymbol{v_c} = [\nu_c, \omega_c]^T$ the vector which defines the velocity of the camera (v_c being the instantaneous linear velocity and ω_c the instantaneous angular velocity), then the relationship between the time derivative of the 2D projection $\dot{\boldsymbol{p}}_{\boldsymbol{img}}$ and the camera velocity $\boldsymbol{v_c}$ is given by

$$\dot{\boldsymbol{p}}_{\boldsymbol{img}} = \boldsymbol{L_x} \boldsymbol{v_c}, \tag{5}$$

where $\boldsymbol{L_x}$ represents the short notation for the interaction matrix $\boldsymbol{L_x}(\boldsymbol{p_{img}}, z_c)$. The complete matrix has dimentions of 2×6, whoever, since the camera is mounted on the AUV, the 4 DOF (instead of six) allows us to simplify it to

$$\boldsymbol{L_x}(\boldsymbol{p_{img}}, z_c) = \begin{bmatrix} -1/z_c & 0 & x_{img}/z_c & -(1+x_{img}^2) \\ 0 & -1/z_c & y_{img}/z_c & -x_{img} y_{img} \end{bmatrix}. \tag{6}$$

IBVS is a form of Image-based control that aims to drive to zero the error between sets of features of the image plane using exclusively movement of the camera. For this work, these features will be keypoint with 2D coordinates.

Let $\boldsymbol{e_{img}}$ be a $2n$ dimensional array which describes the error between the goal features, $s^\star$ and the current image plane features s. These features are a vertical stack of n the keypoint coordinates in the image plane, as such,

$$\boldsymbol{e_{img}} = \boldsymbol{s^\star} - \boldsymbol{s} = \begin{bmatrix} x^\star_{img_1} - x_{img_1} \\ y^\star_{img_1} - y_{img_1} \\ (...) \\ x^\star_{img_n} - x_{img_n} \\ y^\star_{img_n} - y_{img_n} \end{bmatrix}. \tag{7}$$

For this algorithm to be valid, the number of keypoints n selected as features of the image plane should be greater or equal to 3. This error signal represents the relative displacements between the goal keypoints and the current ones. As such, to correct this error, these points should move along these vectors. By considering

$$\dot{\boldsymbol{p}}_{\boldsymbol{img}} = \lambda \left(\boldsymbol{p^\star_{img}} - \boldsymbol{p_{img}} \right), \tag{8}$$

as the desired velocity necessary to minimize $\boldsymbol{e_{img}}$, (5) can be rewritten as

$$\lambda \left(\boldsymbol{p^\star_{img}} - \boldsymbol{p_{img}} \right) = \boldsymbol{L_x} \boldsymbol{v_c}, \tag{9}$$

with λ being a proportional gain to control the overall speed of the process.

Solving (9) with respect to $\boldsymbol{v_c}$ results in a control law for the camera velocity, which moves the current keypoints towards their goal positions, minimizing the error. To achieve this, (9) must be vertically stacked for each of the n selected feature keypoints, resulting in:

$$\lambda \cdot \boldsymbol{e_{img}} = \begin{bmatrix} \boldsymbol{L_{x_1}} \\ \boldsymbol{L_{x_2}} \\ \vdots \\ \boldsymbol{L_{x_n}} \end{bmatrix} \cdot \boldsymbol{v_c}, \tag{10}$$

Let $\boldsymbol{L_x^+}$ be the Moore–Penrose inverse (or pseudo-inverse) of the interaction matrix. Multiplying (10) by $\boldsymbol{L_x^+}$ on the left results in the control law for the camera's velocity, expressed in its own coordinate reference frame. This control law is given by

$$\boldsymbol{v_c} = \lambda \boldsymbol{L_x^+} \boldsymbol{e_{img}}. \tag{11}$$

This camera velocity can then be translated into the vehicle's desired velocity in its own reference frame, $\boldsymbol{v}$, by

$$\boldsymbol{v} = \begin{bmatrix} R_C^B & 0_{3x3} \\ 0_{3x3} & R_C^B \end{bmatrix} \cdot \boldsymbol{v_c}, \tag{12}$$

where R_C^B represents the rotation matrix from the camera's reference frame to the AUV's body reference frame, which is known from the start.

B. Practicalities

The mission is divided into two phases:

1) **Homing phase**, where the goal is to align the AUV with the structure's entrance;
2) **Docking phase**, where the goal is to correctly and safely enter the docking structure.

This entails that for each phase, a distinct set of goal keypoint coordinates must be employed. Moreover, to facilitate the initiation of the homing phase and aid the AUV in locating the docking structure, additional medium to long-range positioning systems should be utilized. Two potential approaches include employing an Ultra-short Baseline (USBL) system with the dock serving as the anchor and transmitting acoustic signals, enabling the AUV to receive relative range and bearing information from the dock; or resurfacing to utilize GPS signals and then integrating that data with Doppler Velocity Log (DVL) measurements as the AUV returns to its operating depth. In the latter method, although the integration of DVL measurements may introduce cumulative position errors over time without a GPS signal, for the brief duration required to

locate and approach the docking structure, the measurements should prove sufficiently reliable.

A challenge arises when the docking structure is not facing the AUV, meaning its entrance is not visible. In the case of the dock used in this thesis, there is only one entrance, although other docking structures may have multiple entrances or even be omnidirectional. For a single entrance dock, solutions to this problem include:

- **Training the neural network with additional detectable keypoints** of the docking structure to recognize the dock from various angles. This would allow the creation of a trajectory to enable the AUV to locate the entrance;
- **Ensuring the docking structure's orientation is fixed and known by the AUV**, allowing the vehicle to move towards its front after completing the previously described positioning procedure using the GPS and DVL.

For this work, the assumption is that one of the aforementioned approaches is employed, ensuring the AUV is always positioned within a cone in front of the dock and never at its back or sides. This assumption eliminates the need to account for scenarios where the dock might be facing away, which would necessitate searching for the entrance. At the beginning of the docking procedure, the AUV will use the USBL information from the dock to orient itself and navigate towards it until the onboard camera detects any of the specified keypoints. Once the structure is recognized in the camera image, the control will transition to full-fledged IBVS. This also assumes a clear path exists between the dock and the vehicle when the command is issued.

C. Interaction Matrix Approximation

The practical implementation of the control law defined in (11) runs into a challenge. As defined in (6), $\boldsymbol{L_x}(\boldsymbol{p_{img}}, z_c)$ is a function of the points depth from the camera focus, z_c. While values for $\boldsymbol{p_{img}}$ can be extracted with a keypoint detecting algorithm running on the camera's images, the values for z_c can't be accurately obtained in real time with the considered setup. To obtain this value, an array of methods could be implemented. One of these is, for example, the use of stereo vision or another range sensor. However, to minimize the equipment on board of the AUV, this value will have to be replaced with an *estimate* $\hat{z_c}$ which will be obtained analytically through inspection of the docking structure during the procedure. This procedure will be further detailed in section VIII. Finally, in other improved robustness towards the depth estimation errors and in order to smooth the transition between current and desired states, the interaction matrix used for the control law was modified to be the model approximation $\hat{\boldsymbol{L_x}}$, defined as

$$\hat{\boldsymbol{L_x}} = \frac{1}{2}\left(\boldsymbol{L_x}(\boldsymbol{p_{img}}, \hat{z_c}) + \boldsymbol{L_x}(\boldsymbol{p^\star_{img}}, z^\star_c)\right), \tag{13}$$

where $z^\star_c$ represents the final desired value for the z_c coordinate of the point in reference to the camera's focus. This control law will converge faster and more robustly than the simpler $\boldsymbol{L_x}(\boldsymbol{p_{img}}, z_c)$, as it as been shown in [4]. That is because using $\boldsymbol{L_x}(\boldsymbol{p_{img}}, z_c)$ requires precise knowledge of the depth z_c. In practice, depth estimation can be noisy and inaccurate. By averaging the interaction matrices computed at the current estimated depth $\hat{z_c}$ and the desired depth z^*_c, the resultant interaction matrix $\hat{\boldsymbol{L_x}}$ becomes less sensitive to these depth estimation errors. Furthermore, the average interaction matrix $\hat{\boldsymbol{L_x}}$ helps to create a smoother transition between the current and desired image features. This can improve the convergence behavior of the visual servoing system, making it more stable and predictable.

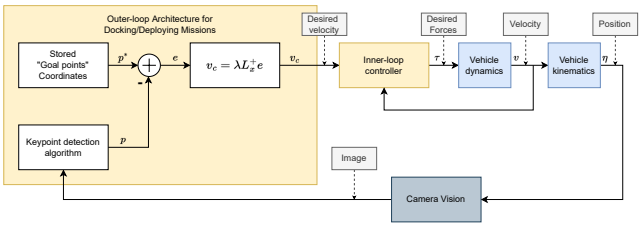

Fig. 3: Complete Control Structure during Docking / Deploying Stage of the mission.

All in all, the complete structure for the outer loop controller, responsible for generating the velocity reference signals is represented in Figure 3

V. Docking Structure

The docking structure used for this work is visible on Figure 4, with the AUV used for comparison. The structure has an upper and lower ramp that act as a funnel to assist the entrance in the cases where it is not perfect. As such, sufficiently slow collisions won't result in failed docking. The entire structure has a bounding box of approximately 1470 x 2014 x 1920 mm. The algorithm and control strategy developed in this work can be adapted to any other type of docking structures.

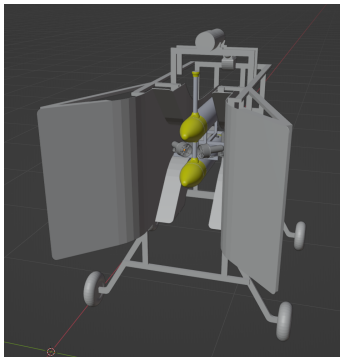

Fig. 4: Medusa Vector inside the Docking Structure.

Table I provides the 3D coordinates of the selected keypoints used for the IBVS controller in the dock's frame of reference, denoted as $\{D\}$. This reference frame is showcased in Figure 5. The more points are selected, the more accurate and resistant to noise the velocities generated by the IBVS algorithm become. Conversely, doing so also increases the challenge of designing a keypoint detection algorithm. In case of a CNN like Yolov8, used for this implementation, more

points require more data acquisition, longer annotation times and more arduous training.

TABLE I: Keypoint 3D coordenates in $\{ D \}$.

ID	X	Y	Z
1	1.4	-0.66	0.43
2	1.4	0.66	0.43
3	1.4	0.66	-0.47
4	1.4	-0.66	-0.47
5	0.188	-0.3	0.62
6	0.188	0.3	0.62
7	0.387	0	0.6815
8	-0.5	-0.25	0.035
9	-0.5	0.25	0.035
10	-0.5	0.25	-0.1
11	-0.5	-0.25	-0.1
12	-0.5	0.0825	-0.1
13	-0.5	-0.0825	-0.1
14	-0.5	-0.0825	0.035
15	-0.5	0.0825	0.035

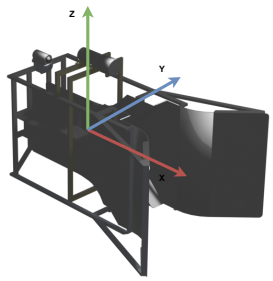

Fig. 5: Adopted reference frame for $\{ D \}$

.

VI. Keypoint's goals

After the selection of the pertinent keypoints in section V, the next step is to obtain the coordinates of these points on the 2D image plane **at the end** of a successful docking operation. This can easily be achieved by calculating these points analytically through the reference frames and models presented thus far. Since the coordinates of the keypoints are known relative to the dock's reference frame {D}, placing the vehicle's reference frame {B} on it's relative target location and adding the camera's reference frame (with known fixed position relative to {B}) allows the use of (4) on these keypoints to directly obtain the theoretical values.

First, using homogeneous coordinates, the transformation of the 3D point coordinates from the docking reference frame, $\boldsymbol{p_D}$, to the camera's reference frame $\boldsymbol{p_C}$ is described as

$$\begin{bmatrix} \boldsymbol{p_C} \\ 1 \end{bmatrix} = \boldsymbol{\Gamma_B^C} \cdot \boldsymbol{\Gamma_D^B} \cdot \begin{bmatrix} \boldsymbol{p_D} \\ 1 \end{bmatrix}, \tag{14}$$

where $\boldsymbol{\Gamma_A^B}$ is the transformation matrix from the frame of reference {A} to {B}, defined as

$$\boldsymbol{\Gamma_A^B} = \begin{bmatrix} \underset{3x3}{\boldsymbol{R_A^B}} & \underset{3x1}{\boldsymbol{T_A^B}} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}, \tag{15}$$

with $\boldsymbol{T_A^B}$ as the translation and $\boldsymbol{R_A^B}$ as the rotation from {A} to {B}. In the case of $\boldsymbol{\Gamma_B^C}$ and $\boldsymbol{\Gamma_D^B}$, for any given known position and orientation of the AUV relative to the dock, the rotations and translations are all known by definition. After acquiring 3D coordinates in the camera's reference frame, $\boldsymbol{p_C}$, obtaining their projection is possible through (4), or, using matrix notation,

$$\begin{bmatrix} \boldsymbol{p_{img}} \\ 1 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} \boldsymbol{p_C} \\ 1 \end{bmatrix}, \tag{16}$$

with

$$\boldsymbol{K} = \begin{bmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{17}$$

with all of the camera intrinsic parameters p_x, p_y, u_0 and v_0 already defined previously. Replacing (14) in (16) gives

$$\begin{bmatrix} \boldsymbol{p_{img}} \\ 1 \end{bmatrix} = \boldsymbol{K} \cdot \boldsymbol{\Gamma_B^C} \cdot \boldsymbol{\Gamma_D^B} \cdot \begin{bmatrix} \boldsymbol{p_D} \\ 1 \end{bmatrix}, \tag{18}$$

which allows the goal coordinates of the projection of any point of Table I to be acquired just by using the vehicles final position and rotation relative to the dock's reference frame.

VII. Keypoint Detection and Yolov8 Model Training

The selected method for keypoint detection utilizes the state-of-the-art YOLOv8 software by Ultralytics [5]. The architecture of the model used for training is called *yolov8n-pose*, which is a generic, adaptable architecture meant to be used for pose estimation tasks provided by Ultralytics. YOLOv8, like its predecessors, is designed to handle varying input sizes due to its fully convolutional architecture. This means it doesn't have fully connected layers at the end, which typically require fixed input sizes. Instead, YOLOv8 can process any input size that's a multiple of 32, allowing it to adapt to different shapes without altering the number of weights. The only real influence on the number of weights comes from the number of classes and how many keypoints we wish to detect. For this work, there are two classes (*dock front* and *dock back*), the first with seven keypoints and the second with eight, for a total of fifteen keypoints, all in Table I. There are, in total, 250 layers and 3127011 parameters.

A custom dataset was created, consisting of 745 images, which were labeled by hand and used to train the neural network. Each image is accompanied by a text file containing the coordinates of specified keypoints in the YOLO pose format.

The trained model can receive an image from the front camera of the AUV and output the positions of the 15 points of the docking structure. If some points are not directly visible or are out of frame, the model will either estimate their positions or fail to identify them.

The training parameters used are specified in Table III, while Table II holds the performance metrics of the model, where

TABLE II: Training parameters for Yolov8

Parameter	Value
Batch Size	4
Max Epochs	5000
Momentum	0.937
Image Size	640
Decay	0.0005
Learning Rate	0.1

TABLE III: Measured metrics for Yolov8

Metric	Value
Precision	96.01%
Recall	98.8%
F1-Score	97.38%

VIII. Camera Depth Estimation

A. Concept

To estimate the value of z_c in real time, an algorithm developed in J. -F. Wu's and M. -Y. Cheng's paper [6] will be implemented. This strategy consists on using the known 3D distances from the keypoints and the captured camera images to create a polynomial expression in function of their depth. Then, by using Newton-Raphson iterative method, its possible to find a close approximation of its root value.

Consider 3 arbitrary points from the keypoint list previously selected, $\boldsymbol{P_1}$, $\boldsymbol{P_2}$ and $\boldsymbol{P_3}$, where $\boldsymbol{P_i} = [x_i, y_i, z_i]^T$. All of coordinates are represented relative to the camera's frame of reference. Since the structure of the dock is well defined, the distances between all of these points are well known numerically. Let l_1 be the distance between P_1 and P_2. This known value can be written as

$$l_1 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}. \tag{19}$$

It is known from the camera model in (4) that

$$\begin{bmatrix} x_{img1} \\ y_{img1} \end{bmatrix} = \begin{bmatrix} \frac{x_1}{z_1} \\ \frac{y_1}{z_1} \end{bmatrix} \tag{20}$$

$$\begin{bmatrix} x_{img2} \\ y_{img2} \end{bmatrix} = \begin{bmatrix} \frac{x_2}{z_2} \\ \frac{y_2}{z_2} \end{bmatrix} \tag{21}$$

which can be rewritten as

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_{img1} \cdot z_1 \\ y_{img1} \cdot z_1 \end{bmatrix} \tag{22}$$

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_{img2} \cdot z_2 \\ y_{img2} \cdot z_2 \end{bmatrix}. \tag{23}$$

Substituting (22) on (19) will yield

$$A_1 z_1^2 + B_1 z_2^2 - D_1 z_1 z_2 - C_1 = 0, \tag{24}$$

with

$$\begin{aligned} A_1 &= x_{img1}^2 + y_{img1}^2 + 1 \\ B_1 &= x_{img2}^2 + y_{img2}^2 + 1 \\ D_1 &= 2(x_{img1}x_{img2} + y_{img1}y_{img2} + 1) \\ C_1 &= l_1^2. \end{aligned} \tag{25}$$

Suppose now that l_2 is the distance between P_2 and P_3, and l_3 is the distance between P_3 and P_1. Similar to the derivation of (24), one will have

$$\begin{cases} g_1(\boldsymbol{Z}) = A_1 z_1^2 + B_1 z_2^2 - D_1 z_1 z_2 - C_1 = 0 \\ g_2(\boldsymbol{Z}) = \mathrm{A}_2 z_2^2 + B_2 z_3^2 - D_2 z_2 z_3 - C_2 = 0 \\ g_3(\boldsymbol{Z}) = A_3 z_1^2 + B_3 z_3^2 - D_3 z_1 z_3 - C_3 = 0 \end{cases}, \tag{26}$$

where

$$\begin{aligned} \boldsymbol{Z} &= [z_1 z_2 z_3]^T \\ A_2 &= x_{img2}^2 + y_{img2}^2 + 1 \\ B_2 &= x_{img3}^2 + y_{img3}^2 + 1 \\ D_2 &= 2(x_{img2}x_{img3} + y_{img2}y_{img3} + 1) \\ C_2 &= l_2^2. \\ A_3 &= x_{img1}^2 + y_{img1}^2 + 1 \\ B_3 &= x_{img3}^2 + y_{img3}^2 + 1 \\ D_3 &= 2(x_{img1}x_{img3} + y_{img1}y_{img3} + 1) \\ C_3 &= l_3^2. \end{aligned} \tag{27}$$

Equation (26) is a system of quadratic equations, which can be solved by a numerical method such as the Newton-Raphson to iteratively find better approximations for the root value Z. Let

$$\boldsymbol{G(Z)} = \begin{bmatrix} g_1(Z) \\ g_2(Z) \\ g_3(Z) \end{bmatrix}, \tag{28}$$

and $\hat{Z}_k$ the root approximation of $G(Z)$ after k iterations of the method. By definition, this value is given iteratively by

$$\boldsymbol{\hat{Z}_{k+1}} = \boldsymbol{\hat{Z}_k} - \boldsymbol{J_G(\hat{Z}_k)^{-1} G(\hat{Z}_k)}, \tag{29}$$

where $\boldsymbol{J_G(\hat{Z}_k)^{-1}}$ refers to the inverse of the 3×3 Jacobian matrix of $\boldsymbol{G(Z)}$, $\boldsymbol{J_G(Z)}$, at the point $\boldsymbol{Z} = \boldsymbol{Z}_k$. By calculating this jacobian at various points and iterations, the depth values of the three points can be estimated.

For the definition of the initial approximation, $\boldsymbol{\hat{Z}_0}$, the assumption that the differences in distance among the depths of three keypoints selected on the dock are small was used. In this case, to obtain a decent initial guess, one can simply assume that the depths of two feature scene points are almost the same, for example,

$$z_1 \approx z_2 \approx \hat{z}. \tag{30}$$

This assumption is even more reasonable in cases where all the keypoints share the same plane. With this assumption, one can describe the displacement of 2D image plane coordinates as a function of $\hat{z}$ and their respective displacement in the 3D world, like such,

$$\begin{cases} \Delta x_{img} = \frac{x_1}{z_1} - \frac{x_2}{z_2} \approx \frac{x_1 - x_2}{\hat{z}} = \frac{\Delta x}{\hat{z}} \\ \Delta y_{img} = \frac{y_1}{z_1} - \frac{y_2}{z_2} \approx \frac{y_1 - y_2}{\hat{z}} = \frac{\Delta y}{\hat{z}} \end{cases}. \tag{31}$$

Recalling (19), it can now be rewritten as

$$l_1 \approx \sqrt{\Delta x^2 + \Delta y^2}, \quad \text{since} \quad \Delta z \approx 0. \tag{32}$$

Consider now the 2D distance between these points in the image plane,

$$l_{img} = \sqrt{\Delta x_{img}^2 + \Delta y_{img}^2}, \tag{33}$$

which when replacing with the values of (31) and (32), wields the following

$$l_{img} \approx \frac{1}{\hat{z}}\sqrt{\Delta x^2 + \Delta y^2} \approx \frac{l_1}{\hat{z}} \Leftrightarrow \tag{34}$$

$$\Leftrightarrow \hat{z} = \frac{l_1}{l_{img}}. \tag{35}$$

This gives us the depth estimation for the first iteration of the Newton Method, $\hat{z} = \frac{l_1}{l_{img}}$.

B. Other details

Once the depth estimation for these three points is obtained, the remaining depths for the other keypoints in the same plane are filled in as the average of the other points. For points of considerably different depths (of different planes), the process is repeated with a new set of 3 selected points. This grouping of points is necessary because the difference in depth between the two sets can be substantial enough to render the approximation in (30) no longer valid in certain situations, which results in the Newton method converging into a wrong approximation. If there are less than 3 points of a given plane available, the depth for those points will not be estimated.

IX. Extended Kalman Filter

In order to minimize this uncertainty and to "store" the evolution of keypoint position even after being covered or incorrectly predicted by the YOLO algorithm, a filtering mechanism is necessary. In this context, a Gaussian Estimation Filter was chosen due to its effectiveness in handling noisy data and providing robust estimates of keypoint positions over time. This section explains its implementation and justifies the choice of the Gaussian Estimation Filter based on its ability to smooth out the uncertainties and maintain consistent tracking of keypoints.

The EKF operates on a nonlinear state space model defined by:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \mathbf{w}_{k-1}, \tag{36}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k, \tag{37}$$

where

- $\mathbf{x}_k$ is the state vector at time step k,
- $\mathbf{u}_{k-1}$ is the control input vector at time step $k-1$,
- $\mathbf{z}_k$ is the measurement vector at time step k,
- $f(\cdot)$ is the nonlinear process model,
- $h(\cdot)$ is the nonlinear measurement model,
- $\mathbf{w}_{k-1}$ is the process noise, assumed to be Gaussian with covariance $\mathbf{Q}_{k-1}$,
- $\mathbf{v}_k$ is the measurement noise, assumed to be Gaussian with covariance $\mathbf{R}_k$.

The EKF algorithm consists of two main steps: prediction and update.

A. Prediction Step

In the prediction step, the state and covariance are propagated through the nonlinear process model:

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}), \tag{38}$$

$$\mathbf{F}_{k-1} = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}}, \tag{39}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}. \tag{40}$$

B. Update Step

In the update step, the state and covariance are updated using the measurement model:

$$\hat{\mathbf{z}}_{k|k-1} = h(\hat{\mathbf{x}}_{k|k-1}), \tag{41}$$

$$\mathbf{H}_k = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k|k-1}}, \tag{42}$$

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k, \tag{43}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}, \tag{44}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}), \tag{45}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1}. \tag{46}$$

C. Defining State and Models

The state we wish to estimate are the positions of a given point in the 2D image plane, $\mathbf{p_{img}} = [x_{img}, y_{img}]^T$ and the point's corresponding depth, Z_c, which is the Z coordinate of the point using the camera's referential. As such,

$$\mathbf{x}_k = \begin{bmatrix} x_{img} \\ y_{img} \\ Z_c \end{bmatrix}_k \tag{47}$$

.

Using a first-order Taylor expansion on the state x_k in (47), the relationship establish in (5) and the assumption that $q \equiv \dot{\theta} \approx 0$, (which is a valid assumption at the operating speeds of this work) results in the process model as

$$\boldsymbol{x}_k = \boldsymbol{A}\boldsymbol{x_{k-1}} + \boldsymbol{B}\boldsymbol{u_k}, \tag{48}$$

where .

$$\mathbf{A} = \mathbf{I}, \tag{49}$$

$$\mathbf{B} = dt \cdot \begin{bmatrix} -1/Z_c & 0 & x_{img}/Z_c & -(1+x_{img}^2) \\ 0 & -1/Z_c & y_{img}/Z_c & -x_{img}y_{img}) \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{50}$$

$$\mathbf{u_k} = \mathbf{v_c}. \tag{51}$$

This **process model** utilizes the same notation as a standard EKF, but it is to note that, due to the nature of the interaction matrix $\boldsymbol{L_x}(\boldsymbol{\dot{p}_{img}}, Z_c)$ being variable based on the specific point being treated, B is non-constant. This is also known as

a State Dependent EKF or State-Dependent Riccati Equation (SDRE) Filter.

The **measurement model** for this filter is an identity observation model or a direct measurement model, meaning that observations $\boldsymbol{z_k}$ have the same structure as the state $\boldsymbol{x_k}$, and as such, the matrix $H = I$.

From (48),

$$f(\boldsymbol{x}_{k-1|k-1}, \boldsymbol{u}_{k-1}) = \boldsymbol{x}_{k-1|k-1} + \boldsymbol{B}\ \boldsymbol{u}_{k-1}, \text{ and} \tag{52}$$

$$\boldsymbol{u}_k = \boldsymbol{v}_c = [v_x, v_y, v_z, p]^T, \tag{53}$$

which, using (50), results in the Jacobian in regards to $\boldsymbol{x}_k$,

$$\mathbf{F}_{k-1} = \begin{bmatrix} \frac{\partial f_1}{\partial x_{img}} & 0 & \frac{\partial f_1}{\partial Z_c} \\ \frac{\partial f_2}{\partial x_{img}} & \frac{\partial f_2}{\partial y_{img}} & \frac{\partial f_2}{\partial Z_c} \\ 0 & 0 & 1 \end{bmatrix}, \tag{54}$$

where F_{k-1} is the state transition matrix and

$$\frac{\partial f_1}{\partial x_{img}} = 1 + dt\left(\frac{v_z}{Z_c} - 2x_{img}p\right), \tag{55}$$

$$\frac{\partial f_1}{\partial Z_c} = \frac{dt}{Z_C^2}\left(v_x - x_{img}v_z\right) \tag{56}$$

$$\frac{\partial f_2}{\partial x_{img}} = -dt\, y_{img}p, \tag{57}$$

$$\frac{\partial f_2}{\partial y_{img}} = 1 + dt\left(\frac{v_z}{Z_c} - x_{img}p\right), \tag{58}$$

$$\frac{\partial f_2}{\partial Z_c} = \frac{dt}{Z_C^2}\left(v_y - y_{img}v_z\right). \tag{59}$$

At each prediction step, when the propagation of the covariance matrix is calculated, the state transition matrix in (54) has its values updated with the respective $\boldsymbol{x}_{k-1}$ and $\boldsymbol{u}_k$ signals.

Furthermore, dt has the value of $0.1\ s$, due to the fact that the input signal $\boldsymbol{u}_k$ is received at a rate of $10\ Hz$.

Although traditional EKF execute prediction and update steps interchangeably, the code developed is asynchronous and event based. This means that, every time an input signal $\boldsymbol{u}_k$ is read, the filter will predict its state (increasing the uncertainty). Likewise, every time a new measurement is received (Yolo Detection + Depth Estimation), the filter will update its state and covariance. Furthermore, if a given point has too high of an uncertainty associated (module of the matrix above a certain threshold), it will not be used by the IBVS algorithm to calculate new reference signals.

In summary, the updated full IBVS system diagram, now including the EKF and Z estimation modules is represented in Figure 6.

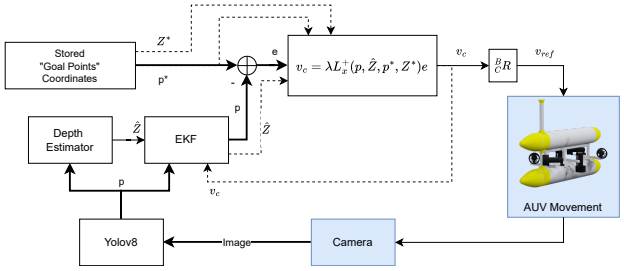

Fig. 6: Completed IBVS architecture.

X. Simulation Results

A. Simplified Keypoint Detection

This first simulation simplifies the general problem by placing markers on the docking structure with a material holding a perfect emissive color value. These markers are perfectly visible and captured by the camera with a constant RGB value, regardless of the environment's lighting conditions. By applying a simple mask on the camera's image, these markers are quickly turned into a 2D region whose center is identified, and its coordinates on the pixel plane are captured. These markers are visible in Figure 7, where the red markers are used for the homing stage while the green ones are used for docking/deploying. This scenario, though not realistic, allows for the assessment of this thesis's strategy independently of the keypoint algorithm's performance.

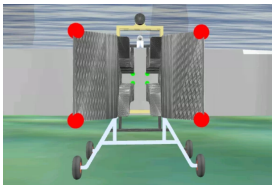

Fig. 7: Dock with perfect emissive color markers.

The following data is taken from one of these simulations, with the starting position being $(4,\ 9.8,\ -1.75)$. Figure 8 shows the AUV's position evolution in 3D and 9 its positional error.

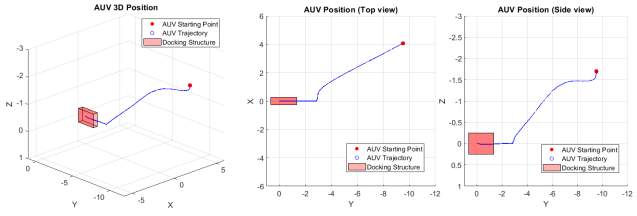

Fig. 8: Evolution of AUV's position during the simplified mission.

The reference tracking performance of the inner-loop controllers is illustrated in Figure 11. The position error stabilized after an initial oscillation, indicating a successful docking operation.

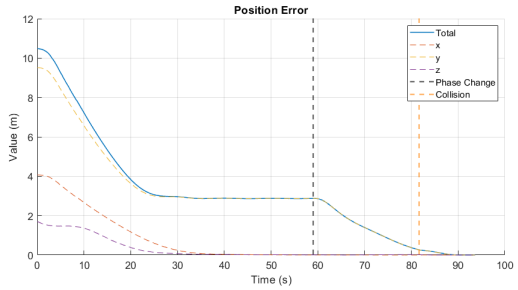

Fig. 9: Evolution of AUV's position error during the simplified mission.

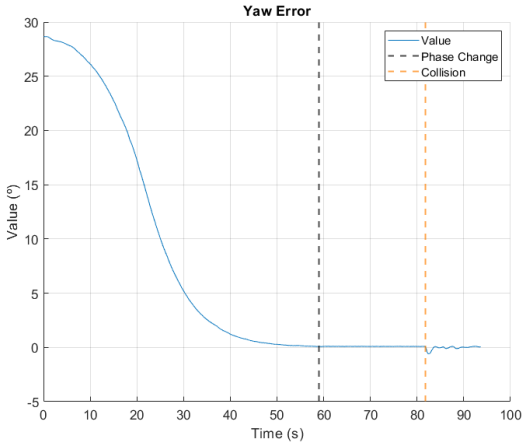

Fig. 10: Evolution of AUV's orientation during the simplified mission.

B. Yolov8 Keypoint Detection, with filtering

For this last mission, no extra markers are present and the keypoints will be detected by Yolov8. Furthermore, the 2D image-plane keypoint coordinates will be filtered by the developed EKF, along with their estimated depth. The evolution of the AUV's position can be observed in Figure 12, along with its position error in Figure 13.

The reference tracking performance of the inner-loop controllers, as illustrated in Figure 15, exhibited very low MSE values. The estimation of the 2D image points by the EKF significantly contributed to generating smoother reference signals, since the Yolov8 model detections were considerably noiser than the simplified scenario. This improvement underscores the efficacy of the EKF in mitigating noise and enhancing the stability of the control system.

XI. Conclusion

The reference signals and error control of the first architecture was the best performing of the 2. That is especially visible when analyzing the AUV positions during the procedure with Figure 9 and Figure 13. The simplified mission converged faster, with smoother reference signals and the error strictly decreasing.

However, these simulations utilized a simplification of the keypoint detection step. Perfectly emissive colorful markings are possible in a simulated environment, but not in real

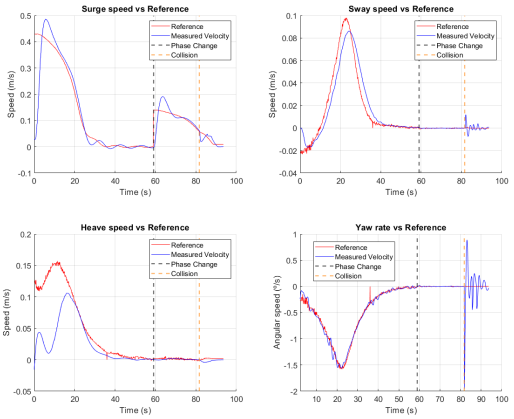

Fig. 11: AUV Inner-controllers reference following during the simplified mission.

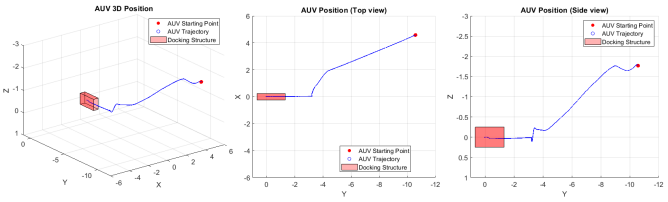

Fig. 12: Evolution of AUV's position during the complete mission.

life. This architecture essentially served to demonstrate that the control tool developed during this thesis are capable of achieving accurate and precise control of the AUV, but their results are ultimately constricted by the keypoint detection chosen.

Yolov8 is a powerful tool, but like any other neural network, it requires extensive and well-labeled data when confronted with complex tasks. The custom dataset utilized in this work was entirely labeled manually. Consequently, small human errors, sometimes as minor as a few pixels when marking keypoint coordinates, will negatively impact the model's training and confuse its validation process. To mitigate this issue,

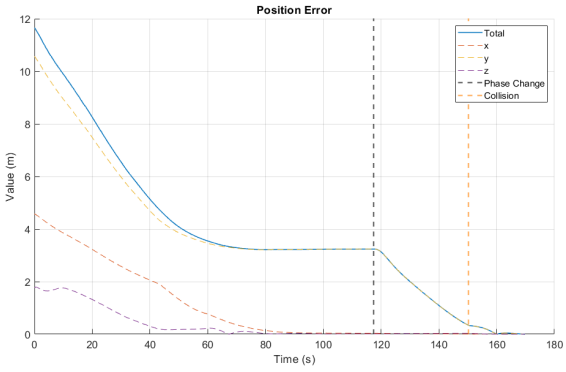

Fig. 13: Evolution of AUV's position during the complete mission.

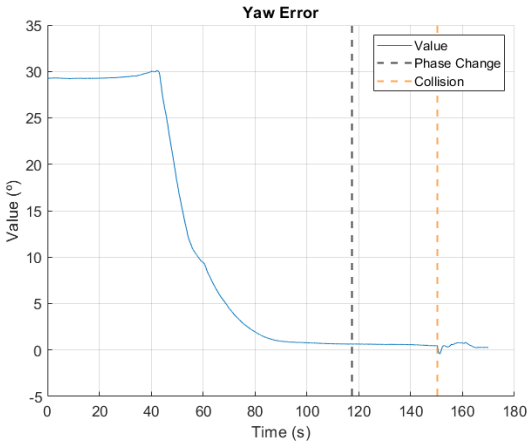

Fig. 14: Evolution of AUV's orientation during the complete mission.

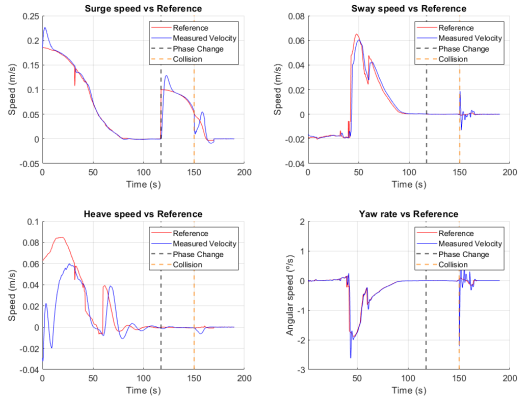

Fig. 15: AUV Inner-controllers reference following during the complete mission.

incorporating more distinct markings on the docking structure would not only assist humans in labeling the data with greater accuracy at the exact coordinates of the keypoints but also provide the CNN with more patterns to recognize.

Depending on the dataset used for training the CNN, this IBVS approach for docking could even be extended beyond artificial docking structures and utilize natural land formations such as tunnels and structures to provide safe protection for the AUV. This could be further enhanced with the inclusion of a few artificial markings, achieving a cost-effective and straightforward mixed approach that combined both artificial and natural methods. The more distinct these keypoints were, the less data was necessary to train the CNN.

References

[1] Organisation for Economic Cooperation and Development (OECD). *The Ocean Economy in 2030*. 2016. doi: https://doi.org/https://doi.org/10.1787/9789264251724-en. URL https://www.oecd-ilibrary.org/content/publication/9789264251724-en.

[2] Diogo Rebelo Teixeira. Sensor-based cooperative control of multiple autonomous marine vehicles. Master's thesis, Instituto Superior Técnico, Lisbon, Portugal, 2019.

[3] David Almeida Souto. Autonomous acquisition and optical transmission of underwater images for online marine habitat mapping. Master's thesis, Instituto Superior Técnico, Lisbon, Portugal, 2022.

[4] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 2, pages 1843–1848 Vol.2, 2004. doi: 10.1109/ROBOT.2004.1308092.

[5] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023. URL https://github.com/ultralytics/ultralytics.

[6] Ju-Feng Wu and Ming-Yang Cheng. Depth estimation of objects with known geometric model for ibvs using an eye-in-hand camera. In *2017 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*, pages 88–93, 2017. doi: 10.1109/ARIS.2017.8297195.