Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Lasse Moen Guttormsen

Underwater Autonomous Grasping of Dead Fish Using a UVMS

Master's thesis in Cybernetics and Robotics
Supervisor: Jan Tommy Gravdahl
Co-supervisor: Bent Haugaløkken, Irja Gravdahl
June 2024

NTNU
Norwegian University of
Science and Technology

Lasse Moen Guttormsen

Underwater Autonomous Grasping of Dead Fish Using a UVMS

Master's thesis in Cybernetics and Robotics
Supervisor: Jan Tommy Gravdahl
Co-supervisor: Bent Haugaløkken, Irja Gravdahl
June 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

Motivated by the occurrence of dead fish and the increasing use of robots within aquaculture, this Master's thesis focused on developing and evaluating control solutions for an underwater vehicle-manipulator system (UVMS) consisting of the BlueROV2 and Reach Alpha 5, specifically targeting the task of grasping dead fish. This is a challenging and delicate operation due to the hydrodynamic effects and the slippery and non-rigid nature of the target.

The research addressed the entire process of grasping dead fish, including detection, pose estimation, convergence, and grasping. A control methodology inspired by the task-priority framework was developed, addressing challenges in task transition with a sigmoid function. Additionally, adaptive tuning was incorporated into the kinematic control law to address changes in the system's situation, ensuring more responsive control in the different parts of the task. A gradient-based optimization approach using the manipulability index was implemented to prevent singularity during the operation.

A 3D visual simulation was developed using Simscape within the Simulink environment. Instead of incorporating the complex dynamic equations of a UVMS, the simulation was structured such that the autonomous underwater vehicle (AUV) was modeled using dynamic equations, with coupling forces from the manipulator introduced as disturbances. The dynamics of the manipulator were constructed using Simscape, leveraging this tool to model the robot arm within the physics engine accurately. The simulation results demonstrated effective convergence, where the UVMS is avoiding undesired collision with the dead fish and utilizing all available DOF to reach a stable grasping point, thus validating the control methodology. The study also evaluated promising object detection methods for detecting and estimating dead fish pose. Additionally, the study explored grasping mechanisms to achieve a stable grasp of the dead fish.

Sammendrag

Motivert av forekomsten av døde fisk og økende bruk av roboter innen akvakultur, fokuserte denne masteroppgaven på utvikling og evaluering av kontroll-løsninger for et undervannsfartøy-manipulator-system (UVMS) bestående av BlueROV2 og Reach Alpha 5, med spesifikt mål om å gripe døde fisk. Dette er en utfordrende og delikat operasjon på grunn av hydrodynamiske effekter og fiskens glatte og bløte egenskaper.

Forskningen adresserte hele prosessen med å gripe døde fisk, inkludert deteksjon, posisjons-estimering, konvergens og gripe-mekanismer. En kontroll-metodikk inspirert av oppgave-prioriteringsrammeverket ble utviklet, og adresserte utfordringer med oppgaveovergang ved hjelp av en sigmoidfunksjon. I tillegg ble adaptiv justering lagt til i den kinematiske kontroll-loven for å takle endringer i systemets situasjon, og sikre mer responsiv kontroll i de ulike delene av oppgaven. For å forhindre singularitet under operasjonen, ble en gradientbasert optimeringsmetode med bruk av manipulerbarhetsindeksen implementert.

En 3D visuell simulering ble utviklet ved bruk av Simscape innenfor Simulink-miljøet. I stedet for å bruke de komplekse dynamiske ligningene til et UVMS, ble simuleringen strukturert slik at det autonome undervannsfartøyet (AUV) ble modellert ved hjelp av dynamiske ligninger, med koblingskrefter fra manipulatoren introdusert som forstyrrelser. Dynamikken til manipulatoren ble konstruert ved hjelp av Simscape, og utnyttet dette verktøyet for å nøyaktig modellere robot armen innenfor fysikk-motoren. Simuleringsresultatene viste effektiv konvergens, der UVMS-en unngår uønskede kollisjoner med den døde fisken og utnytter alle tilgjengelige frihetsgrader for å nå et stabilt gripepunkt, og dermed validerer kontroll-metodikken. Studien evaluerte også lovende objekt-deteksjonsmetoder for å oppdage og estimere døde fiskers posisjon. Studien utforsket også gripe-mekanismer som tar sikte på å oppnå et stabilt grep på død fisk.

Acknowledgement

I would like to express my sincere gratitude to the project advisors for their dedication and presence throughout. Their guidance has been invaluable, steering me on the correct path across various domains and significantly contributing to the results of this research.

I am also grateful to Blue Robotics and Reach Robotics for their responsiveness when I reached out for technical specifications and guidance regarding their products. Additionally, I acknowledge that Appendix A was provided by Reach Robotics, which has been instrumental in constructing the simulation of the arm and enhancing the content and understanding presented in this thesis.

Their collective support has been instrumental in completing this project successfully, and I am profoundly appreciative of their contributions.

Contents

List of Figures

List of Tables

Abbreviations

AUV	Autonomous Underwater Vehicle
CAD	Computer Aided Design
CNN	Convolutional Neural Network
COM	Center of Mass
DOF	Degrees of Freedom
DP	Dynamic Positioning
DPM	Deformable Part Models
FOV	Field of View
INS	Inertial Navigation System
LP	Low-Pass
NED	North-East-Down
PID	Proportional-Integral-Derivative
ROS	Robot Operating System
R-CNN	Region-based Convolutional Neural Network
SDG	Sustainable Development Goals
SMC	Sliding Mode Control
SSD	Single Shot MultiBox Detector
UVMS	Underwater Vehicle-Manipulator System
YOLO	You Only Look Once
YOLOv3	You Only Look Once version 3

1 Introduction

This thesis will present a study and propose a solution to the task of grasping dead fish below the sea surface using a robot arm mounted on an underwater drone. In the course of developing this Master's thesis, it is important to note that certain sections and concepts presented herein have been previously explored in my pre-project [1]. The prior work has provided a solid foundation for the research presented in this thesis. As such, there will be a substantial overlap in content between the two projects in Section 1, 2, 3 and 4. However, it is my intention to expand, refine, and build upon the ideas introduced in the pre-project to offer a more comprehensive and in-depth exploration of the subject matter in this current work.

1.1 Background

In the following sections, a brief background for the task is presented: first, the use of robot manipulation in aquaculture, and second, the problem of dead fish. Thus, the necessary knowledge is provided to justify the research on the specific topic.

1.1.1 Robot Manipulation in Aquaculture

Robotic systems have found valuable applications in the field of aquaculture, particularly in net cleaning, inspection, and monitoring tasks [2], where different solutions exist today. These innovative solutions not only streamline operations but also enhance safety by reducing the risks associated with human divers performing similar tasks. Moreover, the potential for autonomous manipulation in aquaculture presents significant opportunities for cost reduction and improved efficiency. For instance, robots capable of fixing broken nets can save both time and resources by minimizing the chances of fish escaping, limiting the consequences of environmental damage, and reducing the loss of revenue for fisheries. Integrating robotics in aquaculture promises to transform the industry by addressing critical operational needs and promoting sustainability. However, it's worth noting that complex manipulation tasks in underwater environments require ongoing research and development efforts to create robust and reliable robotic designs that can withstand the challenges of aquatic conditions. Therefore, any research in this field will contribute to the efficiency and environmental sustainability of the industry.

1.1.2 The Problem of Dead Fish

In the book *Welfare Indicators for farmed Atlantic salmon: tools for assessing fish welfare* [3] the mortality of the habitats is addressed as;

Mortality has to be recorded on a daily basis. Efficient systems for the collection of dead fish from each tank are a prerequisite for the monitoring of fish performance in aquaculture systems. The increase in the size of tanks and a potential inability to visually observe the bottom of the tanks can prove challenging for the accurate daily registration of dead fish. If possible, the cause of mortality should be noted and dead fish are often preserved for further analysis and inspected by fish health personnel.

This emphasizes the strict regulations on salmon farming regarding dead fish, highlighting the critical need for the industry to implement solutions for managing this challenge.

Furthermore, Figure 1 illustrates the scale of fish mortality in 2021 within MOWI, the biggest Norwegian salmon farmer. The data presented indicate that several locations experienced mortality rates exceeding 30%, demonstrating the significant extent of the problem [4].

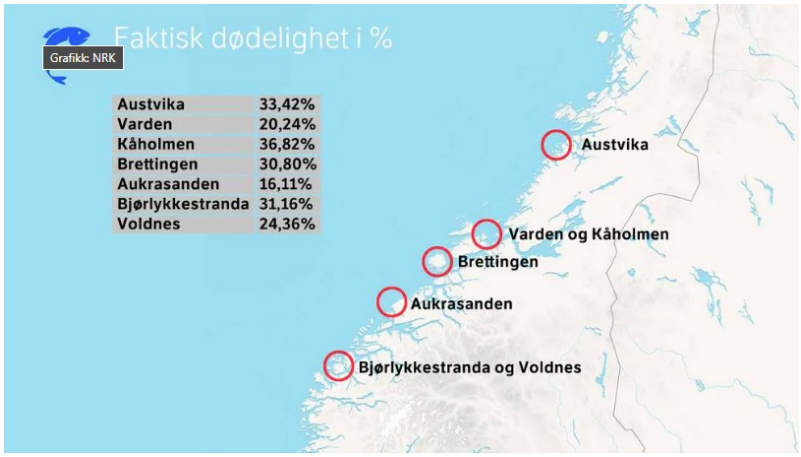


Figure 1: Fish mortality at 7 breeding facilities of the company MOWI [4].

Additionally, during the course of this project, there have been several incidents within the Norwegian aquaculture industry regarding dead fish. For instance, an inspection conducted by the Norwegian Food Authority at one of Lerøy's breeding facilities revealed the processing of dead salmon, a practice that does not comply with regulations. The standards dictate that only live and healthy fish should be prepared for consumer consumption [5]. Another incident regarding the same company was the mass death of salmon across multiple locations [6], shown in Figure 2.

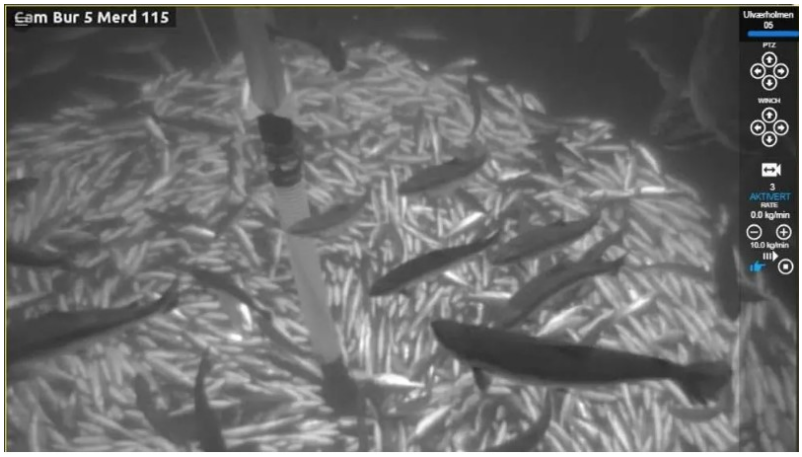


Figure 2: Mass death at one of Lerøy's facilities. Taken from NRK [6].

This accentuates the importance of fish welfare, which can be affected by the presence of dead fish in the cages for an extended duration. Being a potential source of disease and bacteria, the rapid removal of deceased fish is crucial for minimizing the risk of pathogens spreading to healthy fish and water contamination to maintain a healthy aquatic environment [7].

1.2 Problem Formulation

The problem that will be examined in this report is stated as:

Develop an effective robotic solution for grasping dead fish underwater using a multi-link manipulator integrated with an underwater vehicle. Dead fish present a unique grasping challenge due to their flexible and variable shapes in addition to their dynamic behavior while floating. Conventional robotic grasping approaches designed for rigid objects might not be suitable. Developing a methodology that accounts for the variability in fish shapes and ensures a reliable grasp is paramount. Furthermore, underwater environments are notorious for their reduced visibility caused by factors such as water turbidity and lighting conditions. These limitations hinder real-time decision-making and precision manipulation. Designing strategies that enhance the robot's perception capabilities in such environments is crucial for successful fish grasping and manipulation.

Underwater manipulation introduces complex dynamics and interactions between the robotic manipulator, the target object, and the aquatic environment. This complexity necessitates the consideration of numerous factors in the pursuit of an effective solution, which gave rise to a state machine that was developed during the pre-project, illustrated in Figure 3. The task of grasping dead fish encompasses various aspects, prompting the development of a modular approach allowing independent design and development of each step. Although the state machine itself is not directly implemented in the current Master's project, the insights gained from its development have influenced the final proposed solution.

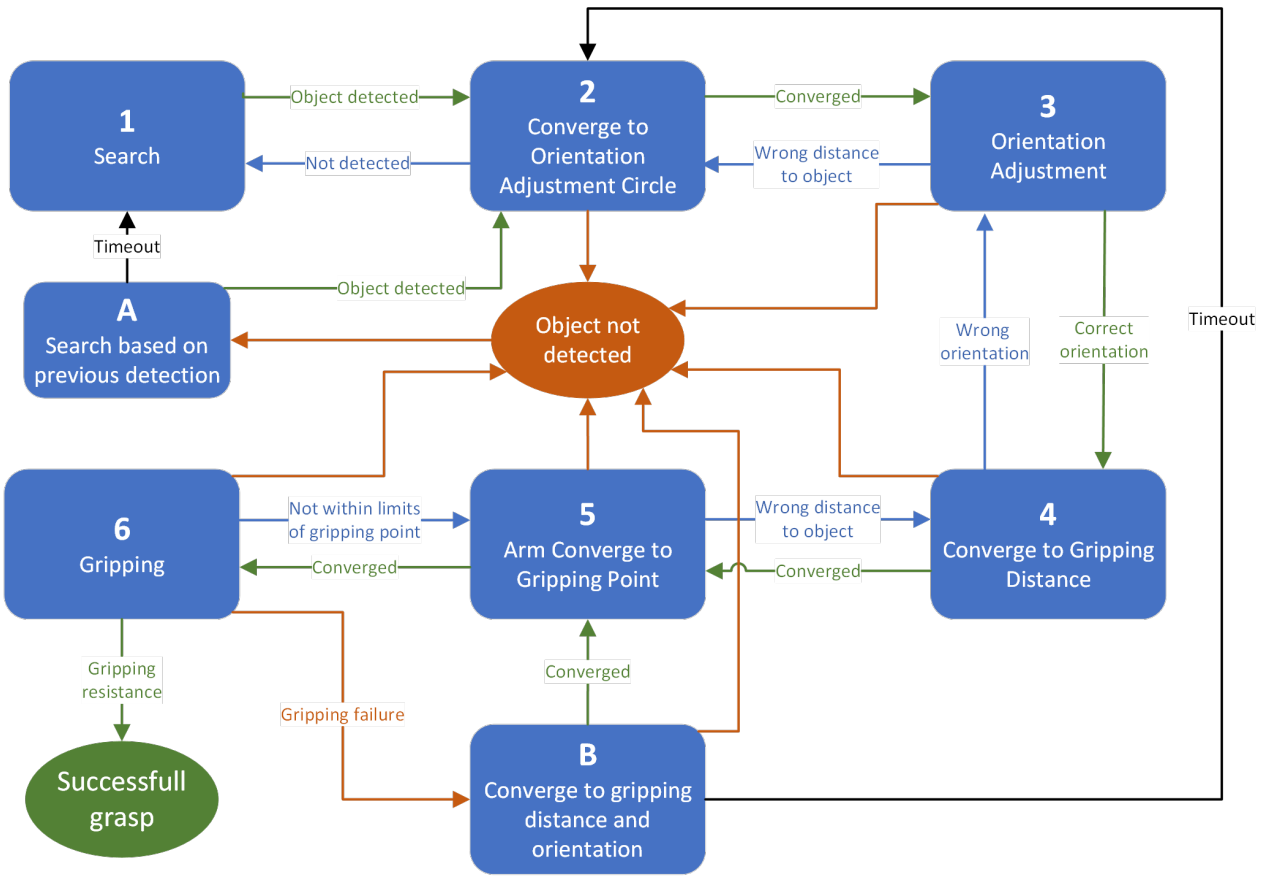


Figure 3: Proposed state machine for the problem.

Building upon these insights, a more concrete problem definition is presented by identifying specific system goals and requirements essential for the development of an effective solution. This refined problem formulation serves as the foundation for targeted research and development efforts.

System Goals

1. **Adaptability to Variability:** The solution must accommodate the wide range of shapes and sizes of dead fish and adapt its grasping mechanism to various cases of the fish's relative position and orientation to the system.
2. **Dynamic Interaction Handling:** Given the unpredictable dynamics of underwater environments, the robotic system must be capable of responding to and compensating for the movements of floating fish and variable currents, ensuring precise manipulation without introducing disturbances to the fish's dynamics.
3. **Secure and Non-Damaging Grasp:** The system must achieve a secure grip on the fish, significantly reducing the chance of slippage while simultaneously ensuring that the grasping mechanism does not inflict damage.
4. **Robustness and Stability:** The control system must exhibit high levels of robustness and stability, ensuring a consistent performance under varying conditions.

1.3 Reach Alpha 5 and BlueROV2

This thesis will look at the decoupled system containing the robotic manipulator Reach Alpha 5 from Reach Robotics and the Autonomous Underwater Vehicle (AUV) BlueROV2 from Blue Robotics, see Figure 4. The Reach Alpha 5 is a fully electric robotic arm with 4-DOF capable of performing various tasks with a variety of different compatible end-effectors. The BlueROV2 is an underwater vehicle equipped with six thrusters mounted such that it results in a maneuverability of 5-DOF. Together, they create a low-cost underwater vehicle manipulator system (UVMS) with good opportunities to carry out various tasks.

(a) Example image of Reach Alpha 5 from Reach Robotics [8].

(b) Example image of BlueROV2 from Blue Robotics [9].

Figure 4: Example images of the hardware that will be examined in this project.

1.4 Delimitations

This thesis focuses on developing a conceptual framework for grasping dead fish using UVMS. However, it's important to delineate the scope of this study by acknowledging certain areas that have not been explored in depth. Firstly, the intricacies of communication with hardware and the direct control of the hardware components have been outside the purview of this research. Such aspects are crucial for real-world applications but were not explored in this project. Additionally, while the pose estimation of the AUV is acknowledged as a fundamental component for precision manipulation, this thesis does not delve into its development or implementation.

In the realm of fish detection and pose estimation, although preliminary research has been conducted to understand the theoretical frameworks and existing methodologies, no practical implementation or testing has been carried out within the context of this thesis. The same holds true for the aspect of gripping. While the challenges of grasping slippery and non-rigid bodies like dead fish have been researched, actual simulation or testing of gripping mechanisms has not been part of this study. These delimitations highlight the focused nature of the thesis on conceptual and theoretical development, leaving room for future work to build upon these foundations with practical implementations and testing.

1.5 Contributions

This thesis proposes a comprehensive solution by integrating existing technologies and methods, enhancing them to create a cohesive system specifically designed for the unique requirements of underwater grasping tasks. Firstly, an adaptation of a sigmoid function to create a seamless state machine. Marey and Chaumette [10] previously used the sigmoid function, but not in the specific context of grasping dead fish. Additionally, an adaptive tuning mechanism for the kinematic control law was developed, adjusting system priorities based on current conditions, thereby enhancing responsiveness and robustness. Furthermore, a gradient-based optimization algorithm, assisted by OpenAI's ChatGPT, was implemented to maintain the manipulability of the robotic arm during the grasping process.

This thesis also introduces a detailed simulation of the UVMS, designed to accurately represent the system dynamics for validating the control strategies. The simulation was constructed using MATLAB Simulink and Simscape, which facilitated the creation of a complex model of the vehicle-manipulator system. This tool has proven instrumental in understanding and optimizing the system's behavior under simulated operational conditions.

1.6 UN Sustainable Development Goals

The application of the UVMS researched in this Master's thesis is particularly relevant to sustainable fish farming, an industry where managing mortality rates is crucial for both economic and environmental sustainability. This technology directly supports the United Nations Sustainable Development Goals (SDGs), specifically SDG 9: Industry, Innovation, and Infrastructure and SDG 14: Life Below Water.

SDG 9 highlights the need for resilient infrastructure and innovation. The integration of sophisticated robotics in fish farming represents a significant advancement in aquaculture

technology, showcasing innovative approaches to problem-solving in industry practices. This project fosters innovation and strengthens the infrastructure of fish farming operations, making them more efficient and sustainable.

SDG 14 emphasizes the conservation and sustainable use of oceans, seas, and marine resources. By deploying the UVMS to efficiently remove deceased fish from aquaculture environments, this project aids in maintaining healthier fish populations and improving water quality, which is essential for the ecosystem's sustainability. Prompt removal of dead fish helps prevent disease spread and decay, which can severely impact marine life and the surrounding environment.

Implementing this technology in aquaculture operations represents a meaningful step toward enhancing sustainability in the industry. It aligns with global efforts to achieve a more sustainable and responsible use of marine resources.

1.7 Report Structure

The remainder of this thesis is organized as follows: Firstly, it will present relevant theory. Section 2 explores the intricacies of modeling and controlling a robot manipulator. Section 3 delves into the domain of robot vision, encompassing algorithms designed to detect and estimate the pose of fish. Section 4 provides insight into modeling vehicle-manipulator systems. Section 5 introduces different control strategies for a UVMS. Following this theoretical groundwork, Section 6 presents the implementation of a created simulation before the implemented control system is presented in Section 7. Section 8 presents the results from the implemented simulation and control system. Subsequently, a detailed discussion of the proposed solution is found in Section 9, supported by the simulation results. Finally, a conclusion of the research is presented in Section 10, summarizing the key findings and implications.

2 Robot Manipulator

In this section, the theory providing the fundamental concepts of robot manipulator modeling and control will be presented. A robot manipulator is a mechanical system designed to perform precise and controlled movements, typically consisting of a series of linked segments or links connected by joints. Figure 5 shows the links (x-, y-, and z-axis frames) of the Reach Alpha 5 manipulator. The figure is from Appendix A, *Kinematic and Dynamic Properties*, which is provided by the manufacturer of the manipulator. It contains relevant information that bridges theoretical concepts with the practical application and operation of Reach Alpha 5.

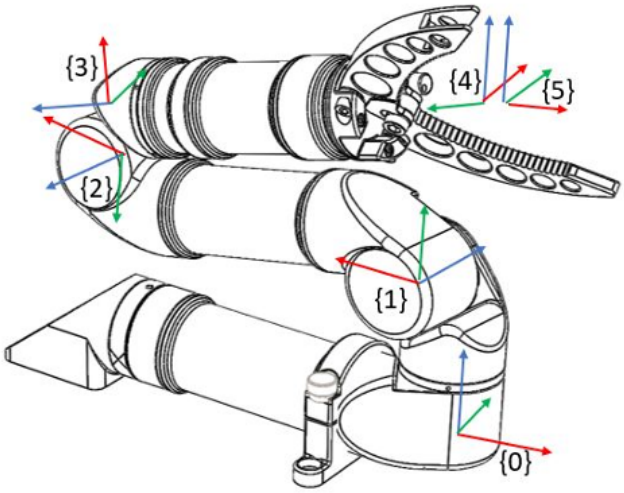


Figure 5: Reach Alpha 5 with its links $\{i\}$. Red is the x-axis, green is the y-axis, and blue is the z-axis. Figure 1 in Appendix A.

The lecture notes *Modelling in Robotics and Control Methods for Robotic Applications* by Freidovich [11] is the main source for the theory explained. This section builds upon the groundwork laid in the pre-project [1].

2.1 Forward Kinematics

Forward kinematics is used to calculate the position and orientation, also known as pose, of an end-effector based on the known joint angles and link lengths of a robot's manipulator. Determining the overall pose of the robot in a specific coordinate system helps in tasks such as robot motion planning, control, and understanding the robot's position in its environment. This section will briefly explain the concept of these calculations found in [11] by looking at kinematic chains and the Denavit-Hartenberg convention.

2.1.1 Kinematic Chains

A robot manipulator is composed of a set of rigid bodies connected together by joints. These rigid bodies are often referred to as links, and they represent specific positions and orientations within the world coordinate frame, as seen in Figure 5. As for the joints,

there exist different types which create different types of motion. Most common are the prismatic and revolute joints creating linear and angular motion, each providing a single DOF. Additionally, there exist multi-DOF joints, such as spherical joints, which provide a broader range of motion. This report will not delve into those, as the Reach Alpha 5 manipulator consists of only revolute joints. When combined, these links and joints form a cohesive representation of the manipulator's overall motion, commonly referred to as a kinematic chain. Figure 6 illustrates the links and joints of the Reach Alpha 5 and provides a visual aid for the following theory.

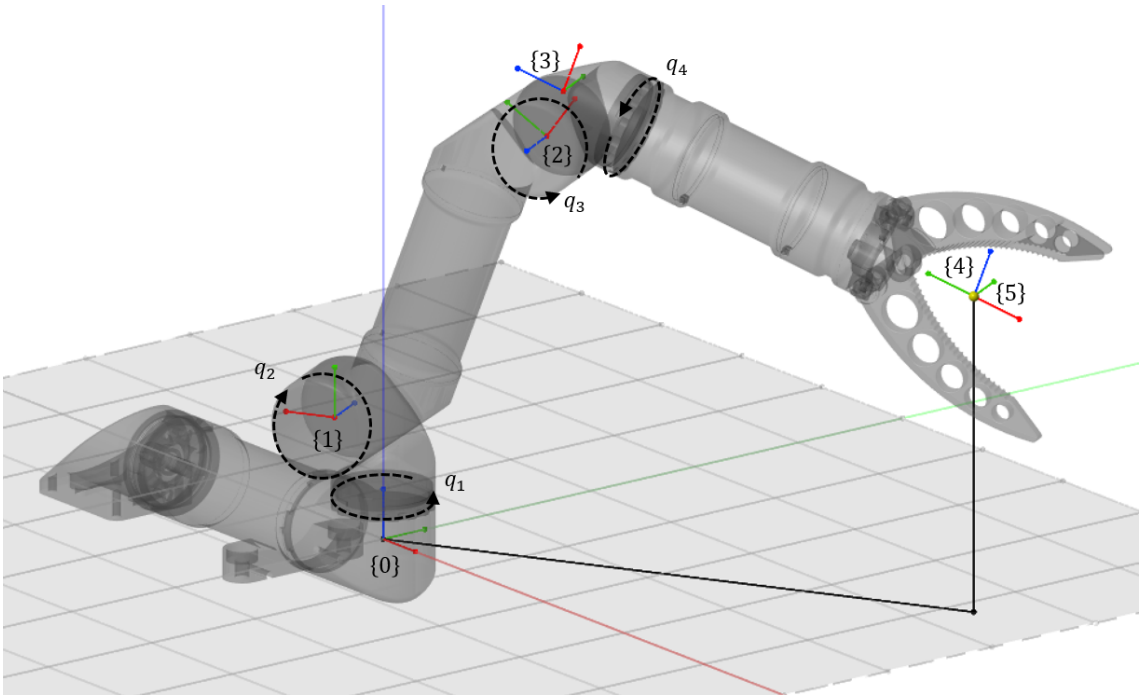


Figure 6: Reach Alpha 5 with its joints, q_i, and links, $\{i\}$.

To ensure uniformity and facilitate easy referencing, a widely recognized convention is employed for numbering the links and joints in robot manipulators. According to this convention, joints are sequentially numbered, beginning with 1. Consequently, each joint i serves as the connection point between link (i-1) and link i [11]. Hence, in a robot manipulator with i joints, the total number of links will be i+1. The actuation of joint i induces motion in link i. Additionally, these joints are typically represented by the variables q_i, denoting their respective joint angles or positions. For the two types of single-degree-of-freedom joints, q_i is defined as follows,

$$q_i = \begin{cases} d_i, & \text{if joint } i \text{ is prismatic} \\ \theta_i, & \text{if joint } i \text{ is revolute} \end{cases}, \tag{1}$$

where d_i denotes displacement along the axis of actuation by joint i, and θ_i denotes the angle of rotation in joint i [12].

To further analyze how motion in each joint affects the manipulator's kinematic chain, each link is rigidly attached to a frame i that represents the pose of the rigid body. Any point on link i will therefore remain constant in the i^{th} frame. Therefore, motion in joint i that changes the pose of link i will consequently change the pose of frame i. The relation between the frames is described using a *homogenous transformation matrix* of the form

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{o} \\ 0 & 1 \end{bmatrix}, \tag{2}$$

where $\mathbf{R}$ is a rotation matrix describing the orientation of the frame and $\mathbf{o}$ is a vector describing the translation in Cartesian coordinates of the origin. Such that each frame can be computed from the previous frame and the motion of the connected joint. This transformation matrix is denoted $\mathbf{A}_i = \mathbf{A}_i(q_i)$ such that the transformation matrix from link i to j can be computed as [11]

$$\mathbf{H}_j^i = \begin{cases} \mathbf{A}_{i+1}\mathbf{A}_{i+2}\cdots\mathbf{A}_{j-1}\mathbf{A}_j, & \text{if } i < j \\ \mathbf{I}, & \text{if } i = j \end{cases}. \tag{3}$$

This transformation matrix enables the mapping of any point on the manipulator to the world frame, thereby determining the pose of the end-effector based on the specified joint values. Given the potential complexity of kinematic links in advanced manipulators, the following section will detail a widely adopted convention for positioning these links. Adopting this convention simplifies the computation of forward kinematics, lending structure to the process and reducing complexity.

2.1.2 The Denavit-Hartenberg Convention

The transformation between two frames can be defined by 6 parameters, 3 for the relative position of the origin and 3 for the relative orientation of the axes [11]. However, if the links of the robot manipulator are structured more systematically, the number of transformations may be reduced. The Denavit-Hartenberg (DH) convention is such a method where the transformation $\mathbf{A}_i$ can be constructed with only 4 parameters,

$$\begin{aligned} \mathbf{A}_i &= \text{Rot}\,(z,\theta_i)\,\text{Trans}\,(z,d_i)\,\text{Trans}\,(x,a_i)\,\text{Rot}\,(x,\alpha_i) \\ &= \begin{bmatrix} \mathbf{R}_{z,\theta_i} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & \mathbf{o}_{z,d_i} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & \mathbf{o}_{x,a_i} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{x,\alpha_i} & 0 \\ 0 & 1 \end{bmatrix}. \end{aligned} \tag{4}$$

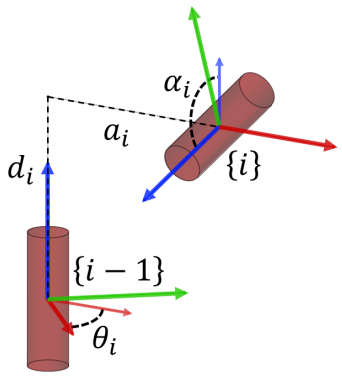


Figure 7: Illustration of the 4 parameters of the DH convention.

This systematic approach starts by assigning coordinate frames to each link such that the z-axis aligns with the rotation axis of any corresponding revolute joint and the x-axis

aligns with the translation axis of any prismatic joint. Additionally, two restrictions must be upheld for the reduction in parameters to be possible [12],

- The axis $\mathbf{x}_i$ is perpendicular to $\mathbf{z}_{i-1}$
- The axis $\mathbf{x}_i$ intersects the axis $\mathbf{z}_{i-1}$

This initial frame assigning ensures a consistent basis for subsequent transformations. When executing the 4 parameters, it is important to recognize that with each step, the reference frame is adjusted. This means that each transformation is applied relative to the newly adjusted frame from the previous step. The parameter execution can be structured as follows,

1. Translation along the z-axis.
2. Rotation around the z-axis that includes the potential joint's angle (θ_i) and any additional predefined rotation.
3. Translation along the x-axis that includes the potential prismatic joint translation (α_i) and any additional predefined translation.
4. Rotation around the x-axis.

These parameters can be structured in a table referred to as the DH-parameters. The DH-parameters for the Reach Alpha 5 can be found in Table 1 in Appendix A, but from Figure 5, the numbering is wrong. Therefore it is restructured in Table 1, where link 0 is aligned with the world frame and the joint numbering, θ_i, starts from 1.

Table 1: Restructured DH-parameters for Reach Alpha 5. $\theta_a = \tan^{-1}(\frac{145.3}{40})$.

Link	**d** [mm]	$\boldsymbol{\theta}$ [rad]	**a** [mm]	$\boldsymbol{\alpha}$ [rad]
1	46.2	$\theta_1 + \pi$	20	$\frac{\pi}{2}$
2	0	$\theta_2 - \theta_a$	150.71	π
3	0	$\theta_3 - \theta_a$	20	$-\frac{\pi}{2}$
4	-180	$\theta_4 + \frac{\pi}{2}$	0	$\frac{\pi}{2}$
5	0	$-\frac{\pi}{2}$	0	0

Let's look at the transformation from link 0 to link 1 using Figure 8 for visual aid. First, the origin is translated with $d_1 = 46.2mm$ along the z-axis, before a rotation of π radians around the z-axis, given that θ_1 is 0. Furthermore, the origin is translated along the new x-axis with $a_1 = 20mm$ and finally rotated $\frac{\pi}{2}$ radians around the x-axis to get the frame of link 1. In Figure 8, $\theta_2 = \frac{\pi}{2}$ because with all joint angles equal 0, the robot arm is in a configuration resulting in self-collision. But as an example, we get $\boldsymbol{\theta}_2 \approx 15.4°$ corresponding with the translation direction of a_2 relative to link 1.

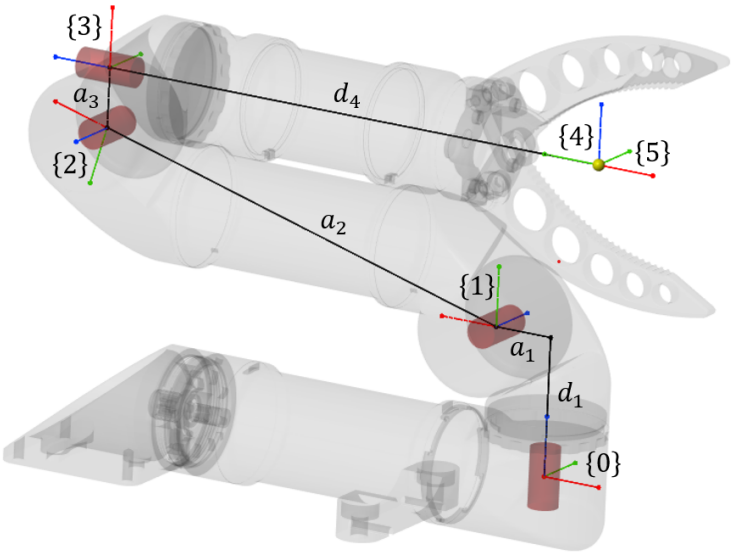


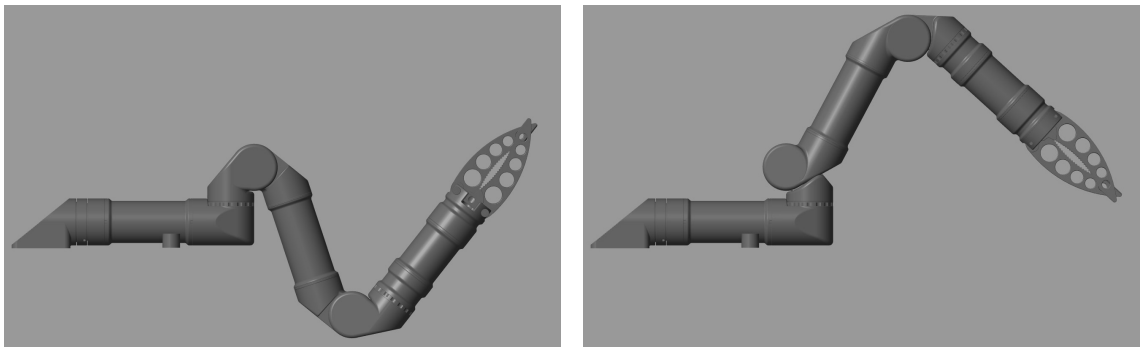
Figure 8: Visualization of the DH-parameterisation for Reach Alpha 5. $\theta_{1,3,4} = 0$ and $\theta_2 = \frac{pi}{2}$.

2.2 Inverse Kinematics

While forward kinematics computes the pose of the different frames of the robot manipulator with the given joint angle, inverse kinematics computes the joint angles to achieve the desired pose of the end-effector. The objective is to find a set of joint angles $(q_1, q_2, ..., q_n)$ that satisfy the equation [11]

$$\mathbf{H}_n^0(q_1, \cdots, q_n) = \mathbf{A}_1(q_1)\mathbf{A}_2(q_2)\cdots\mathbf{A}_n(q_n) = \mathbf{H}, \tag{5}$$

where n is the number of joints. As the number of joints increases, the inverse kinematics problem becomes more complex, potentially yielding multiple configurations that achieve the same end-effector pose. This gives rise to the terms elbow down and elbow up illustrated in Figure 9 where both configurations will result in the same end-effector position.



(a) Elbow down.

(b) Elbow up.

Figure 9: Illustrating how two configurations can lead to the same goal position.

Note that the orientation will not be equal in this case because the Reach Alpha 5 is

constrained by the number of DOF. In contrast, other manipulators possessing additional DOFs can attain the same end-effector orientation and position through various joint configurations. To address these challenges, the inverse kinematics problem is often decomposed into the inverse position problem and the inverse orientation problem. Tackling these problems independently can sometimes simplify the overall solution process, although success is not guaranteed. The division is also practical from a design perspective, influencing the construction of robots to favor architectures that facilitate such decomposition. Additionally, certain applications may only necessitate determining the end-effector's position, rendering the orientation aspect irrelevant [11][12].

For manipulators with few joints the inverse kinematic problem may be solved geometrically instead of using equation (5) [12]. With Reach Alpha 5 as an example, Figures 3, 4, 5, and 6 in Appendix A illustrate how the inverse kinematic equations can be solved geometrically, both for elbow down and up configuration.

2.3 Inverse Velocity Kinematics

While inverse kinematics finds a desired pose of the end-effector, inverse velocity kinematics finds a desired linear and angular velocity of the end-effector. The choice between these two approaches hinges on the specific task requirements and the desired motion for the manipulator. In the context of a robot arm connected to an AUV, they create a decoupled system, where the use of velocity control becomes particularly advantageous. A more detailed discussion on this topic will be presented in Section 5. This section will include the calculation of the Jacobian matrix for the robot manipulator and desired joint velocities.

2.3.1 Manipulator Jacobian

The Jacobian matrix is a mathematical construct that encapsulates how the motion of an end-effector is influenced by the velocities of its individual joints q_n. It can be divided in two parts $\mathbf{J}_v$ and $\mathbf{J}_\omega$ such that [11]

$$\mathbf{v}_n^0(t) = \mathbf{J}_v(q(t)) \cdot \dot{\mathbf{q}}(t) \quad \text{and} \quad \boldsymbol{\omega}_n^0(t) = \mathbf{J}_\omega(q(t)) \cdot \dot{\mathbf{q}}(t). \tag{6}$$

Combining them gives the manipulator Jacobian $\mathbf{J}(\mathbf{q}(t)) = [\mathbf{J}_v(\mathbf{q}(t))\ \mathbf{J}_\omega(\mathbf{q}(t))]^T$ which can be used to calculate the linear and angular velocity of the end-effector

$$\boldsymbol{\xi}(t) = \begin{bmatrix} \mathbf{v}_n^0(t) \\ \boldsymbol{\omega}_n^0(t) \end{bmatrix} = \mathbf{J}(\mathbf{q}(t))\dot{\mathbf{q}}(t). \tag{7}$$

The size of the Jacobian for a manipulator arm with n number of joints is $6 \times n$ such that each column represents the linear and angular velocity contribution from each joint. The Reach Alpha 5 will, therefore, have a Jacobian matrix of size 6×4 due to its 4 joints.

The Jacobian matrix can be computed using the DH-parameters. Let's again divide the Jacobian into its two parts, $\mathbf{J}_v$ and $\mathbf{J}_\omega$. Firstly, the linear velocity of the end-effector in a manipulator arm consisting of only revolute joints will be a result of the angular velocity of the joint and its distance from the center of rotation. Thus, each column i can be found as

$$\mathbf{J}_{v_i} = \mathbf{z}_{i-1}^0 \times [\mathbf{o}_n^0 - \mathbf{o}_{i-1}^0], \tag{8}$$

where $\mathbf{z}_{i-1}^0$ is the z-axis of frame $i-1$ represented in world frame and $\boldsymbol{o}$ is the position of the frame origin. Secondly, the angular velocity of the end-effector can be computed as a sum of the rotations from each joint. Therefore $\mathbf{J}_\omega$ needs to include information about the direction of the angular velocity giving,

$$\mathbf{J}_{\omega_i} = \mathbf{z}_{i-1}^0. \tag{9}$$

2.3.2 Joint Velocities

Desired joint velocities can simply be calculated by rearranging equation (7) with respect to $\dot{\mathbf{q}}(t)$ using the inverse of the Jacobian such that

$$\dot{\mathbf{q}}(t) = \mathbf{J}(\mathbf{q}(t))^{-1}\boldsymbol{\xi}(t). \tag{10}$$

However, it is important to note that this approach is applicable only when the Jacobian is square and invertible, making it unsuitable for situations when the robot finds itself in a singular configuration. In such cases, there exist other alternative methods for assigning joint velocities by using the pseudo inverse and weighting matrices, which will be further discussed in Section 5.1.

2.4 Workspace and Manipulability

Both workspace and manipulability are important to evaluate to increase the robot's ability to perform a given task. The workspace of a robotic arm refers to the three-dimensional region or space where the end-effector can reach and operate, while the robot arm avoids self-collision. The workspace of Reach Alpha 5 can be found in Appendix A, but this will change when mounted on the AUV due to the risk of colliding with its frame. Additionally, manipulability can be analyzed, which is a measurement of how effectively a robotic arm can move its end-effector in each DOF from its current position. It also indicates how close the manipulator is to a singular configuration.

For the problem at hand, the possibility of positioning the AUV such that the fish is within the workspace and the manipulability of the robot arm is optimized, will be beneficial when performing the grasping of dead fish. Therefore methods of quantifying the manipulability and how to evaluate the workspace will be presented.

2.4.1 Manipulability Index and Ellipsoids

Yoshikawa introduced the manipulability index, which is a scalar that describes the distance to singular configurations [13]. The manipulability measure is defined as

$$w = \sqrt{\det(\mathbf{J}\mathbf{J}^T)} \tag{11}$$

and provides an instant evaluation of the manipulability that the manipulator has at the current configuration. In order to get more detailed information about manipulability,

one can examine the manipulability ellipsoids, which is a representation of the possible velocities of the end-effector in all directions at a current configuration. The ellipsoid is found by looking at the eigenvalues λ_i and eigenvectors $\mathbf{v}_i$ of the matrix

$$\mathbf{A} = \mathbf{J}\mathbf{J}^T \tag{12}$$

such that the vectors spanning the ellipsoid in Figure 10 can be found as

$$\mathbf{a} = \sqrt{\lambda_1}\mathbf{v}_1 \tag{13a}$$
$$\mathbf{b} = \sqrt{\lambda_2}\mathbf{v}_2 \tag{13b}$$
$$\mathbf{c} = \sqrt{\lambda_3}\mathbf{v}_3 \tag{13c}$$

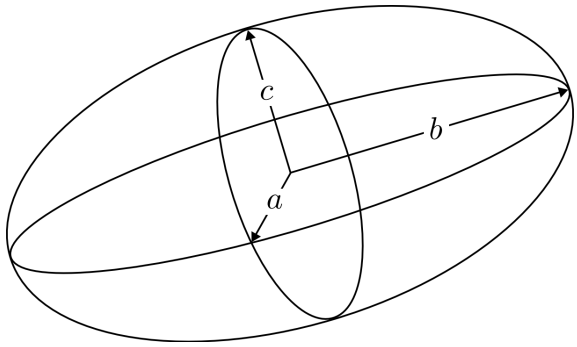


Figure 10: Example of a manipulability ellipsoid spanned from the vectors **a**, **b** and **c** created by the eigenvalues and eigenvectors of **A**.

2.4.2 Joint Limits

Another aspect of workspace and manipulability is the joint limits of the robotic arm. This is defined based on the specification of the actuators and by avoiding self-collision. The reachable workspace of the Reach Alpha 5 is found in Appendix A, but when mounted on the AUV it will create additional space where collision with the AUV might occur, thus changing the workspace. This is an important note that needs to be considered when creating simulations and testing with the actual system.

2.5 Manipulation

In the realm of robotics, the capability to manipulate objects effectively is as critical as the robot's ability to perceive and navigate its environment. Manipulation involves the mechanical interaction between a robot and the objects it handles, primarily facilitated through its end-effector. This section delves into the intricacies of robotic manipulation, focusing specifically on the challenges and solutions related to grasping dead fish, which is non-rigid and slippery. Such objects pose unique challenges due to their tendency to deform or slip under grip pressure, necessitating advanced strategies for secure and stable manipulation.

2.5.1 Grasping

There are multiple solutions to grasping an object. This section will look into form closure and force closure, which represent two ways to achieve a stable grasp. The book, *Springer Handbook of Robotics* [14], is the main source for this theory.

Form Closure
Form closure is achieved when the contact points between the gripper and the object mechanically constrain the object, preventing any movement in any direction. This property does not depend on the force applied by the gripper but rather on the geometric configuration of the contact points. It is particularly useful when the object's surface properties do not provide good friction or when the object must not be squeezed with high forces, as in the case of delicate or brittle materials. Figure 11 shows three cases where the left and middle cases are valid form closure and the right case is not. The object is not mechanically constrained in the right case, as a slight rotation would allow it to escape the grasp.

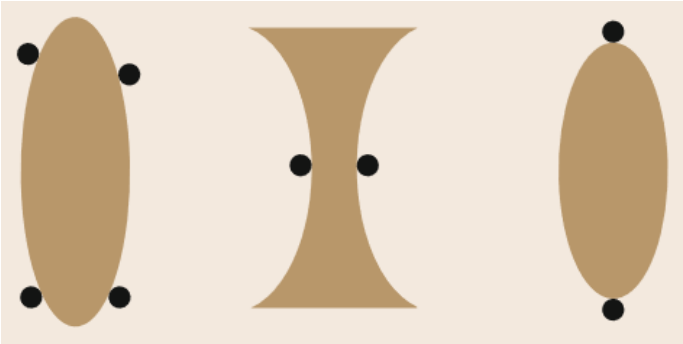


Figure 11: Examples of two dimensional grasp. Two with form closure of different orders and one without form closure [14].

With form closure, the presence of multiple viable and stable solutions may be possible. The left case in Figure 11 would be valid even though the contact points were slightly adjusted. Therefore it necessitates the utilization of optimization algorithms, determining suitable contact points based on task constraints. Optimization can also be used to find the minimum amount of contact points needed to achieve form closure. [14]

Force Closure
Force closure, in contrast, depends on the forces applied at the contact points to secure the object. This method ensures that the object cannot escape the grip due to the applied forces creating sufficient friction. Force closure is applicable in situations where form closure is not possible due to the object's shape or other constraints. It is essential in handling objects that are slippery or those that have surfaces that do not easily conform to the gripper's geometry.

One common definition of force closure is to allow each contact force to lie in its friction cone rather than along the contact normal, referred to as frictional form closure [14]. The friction cone represents the range of potential tangential forces that can be applied at a contact point without causing sliding or slipping. Figure 12 shows an example of a friction cone. If the angle of the applied contact force, β, exceeds the threshold at which the tangential component, μf_z, surpasses the maximum frictional force, slippage will occur.

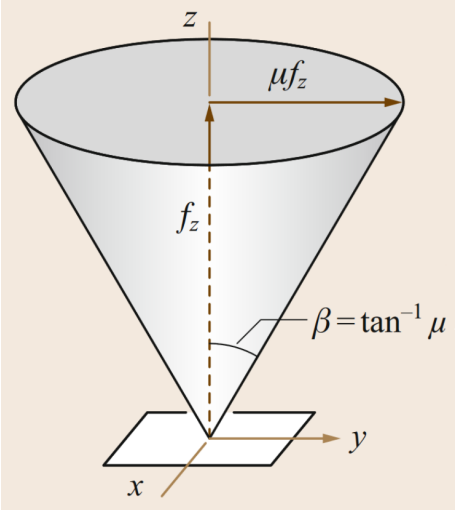


Figure 12: Example of a friction cone [14].

Form closure offers significant advantages for grasping dead fish due to their non-rigid and slippery nature, mitigating the risk of damage which is important for subsequent analysis. However, limitations stemming from gripper design may impede achieving form closure, particularly when insufficient contact points are available to adequately constrain the fish's shape. In such cases, force closure becomes essential, necessitating precise control of gripping force to safeguard the integrity of the dead fish.

2.5.2 End-Effector

The Reach Alpha 5 can be equipped with multiple different grasping tools or end-effectors, but the current configuration is a simple single-interlock jaws shown in Figure 13. The gripper is actuated by a rod pushing and pulling the bolt connecting the two jaws. There is no sensors that can measure force or slip, but the gripper has feedback of the gripper opening and a camera monitoring the gripper. Understanding the capabilities and limitations of the existing system is vital to make an assesment as to whether the end-effector of the Reach Alpha 5 should be changed for the task of grasping dead fish. Therefore a brief research of alternitive gripper designs have been conducted.

Figure 13: Standard jaws for Reach Alpha 5 [8].

A variety of gripper designs are available for different manipulation tasks. Baohua Zhang [15] provides a detailed summary of state-of-the-art grasping technologies and sensor-based control methods that are particularly applicable to agriculture and food industries, where objects often share similar characteristics with dead fish. For instance, Zhang discusses various sensors integrated into grippers that enhance their flexibility and control, such as tactile sensors. Tactile sensors can measure tactile-related properties like force and slip, creating the possibility of monitoring and controlling the grasping process in more detail. However, this type of technology is expensive and adds to the complexity. Therefore, specific gripper technologies such as soft and flexible grippers present an intriguing alternative for the task at hand. With a design that adapts to the shapes of the grasped object, reducing damage while creating a secure grasp. Figure 14 shows some of the interesting designs mentioned by Zhang [15].

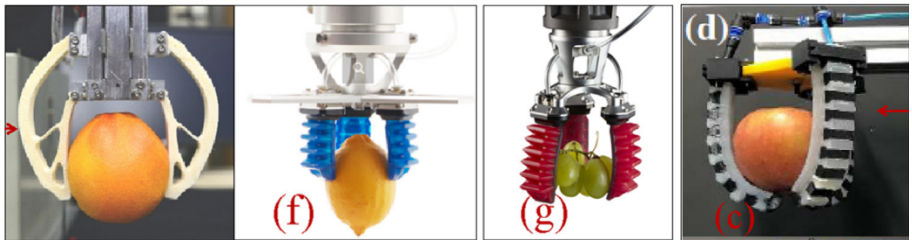


Figure 14: Different gripper designs for agriculture [15].

The design of the end-effector is a crucial factor that affects the complexity of the task at hand. There are many different designs and functionalities created for all sorts of different manipulation problems, and a simple jaw gripper, which operates without any form of direct feedback, is not the ideal choice for the specific task of grasping dead fish. Nevertheless, it is important to explore the limits of what can be achieved with a minimalist manipulator. This pursuit enables the development of resilient control systems and contributes to the adaptability and versatility of robotic systems. With a single robotic arm capable of multiple tasks, industries benefit from cost-efficiency, as they can execute a range of autonomous operations using a single integrated system within their facilities.

3 Robot Vision

To successfully grasp a dead fish, it is essential first to detect it and estimate its pose relative to the UVMS. The system currently utilizes monocular cameras, one mounted on the AUV and another on the robot manipulator. Camera systems are critical for enhancing situational awareness and providing insights during specific underwater operations. While acoustic sonars are commonly used in underwater environments, as LiDAR does not function underwater, this system does not include such sensors. Therefore, this section will explore potential solutions for detecting dead fish and estimating its pose using camera systems, delving into relevant theories and techniques on object detection and pose estimation.

This section provides a brief overview of the topics and its intention is to create an introduction to the challenges that may arise for future investigations. This builds upon the groundwork laid in the pre-project [1].

3.1 Object Detection

Multiple algorithms for object detection exist, for example, Single Shot Detector (SSD), Deformable Part Models (DPM), and Region-based Convolutional Neural Network (R-CNN). In 2016 a new method based on neural networks was developed by Joseph Redmon et al. [16] called You Only Look Once (YOLO). This method outperformed other approaches and has become a new standard in object detection. In 2018, they presented an upgraded version, YOLOv3, with similar accuracy to SSD but three times faster [17]. YOLOv3 has already been used in underwater object detection, similar to this project of grasping dead fish. Firstly, Haugaløkken et al. [18] used the BlueROV2 with the robotic arm SeaArm to grasp a known object using YOLOv3 for object detection. Secondly, Ming En Koh et al. [19] proposed a solution for fish pose estimate by using the bounding boxes for both the body and head of the fish found with YOLOv3, which will be further discussed in Section 3.2.

YOLO

YOLO performs object detection in real-time by analyzing entire images in one evaluation, hence the name You Only Look Once. It applies a single convolutional neural network (CNN) to the full image, which makes it exceptionally fast. Figure 15 illustrates the flow of the algorithm. First, it divides the image into a grid. Each grid cell predicts a certain number of bounding boxes, with a confidence score [20]

$$p_1 = P(\text{Object}) \cdot \text{IoU} \tag{14}$$

where IoU is the *intersect over union* between the predicted box and the ground truth. Thus it indicates both the confidence that the box contains an object and the precision of the box's placement. Furthermore, it creates class probability for each cell [20]

$$p_2 = P(\text{Class}|\text{Object}) \tag{15}$$

which are conditional on that cell containing an object. This results in the class probability map in Figure 15. Finally, the class-specific score for each bounding box is found as

$$p = p_1 * p_2 = P(\text{Object}) \cdot \text{IoU} * P(\text{Class}|\text{Object}) \tag{16}$$

Once this probability is determined, a threshold can be established to filter and retain only the most probable results as output [16].

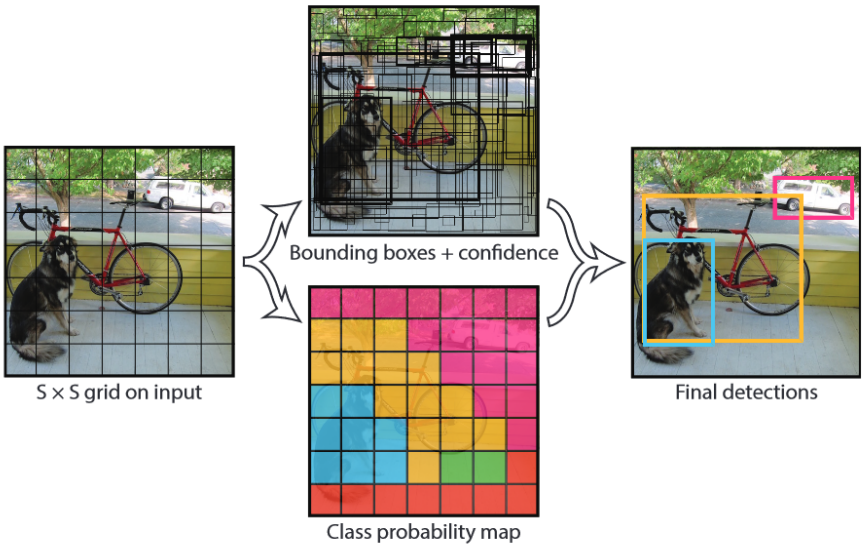


Figure 15: YOLO architectural flow to determine bounding boxes for different objects [16].

Neural networks rely on training to detect various desired objects. There are multiple datasets available for this purpose, and pre-trained models can be found as open-source resources. However, it is not confirmed that there are models specifically trained for detecting farmed salmon.

3.2 Estimate of Fish Pose

The fish pose consists of both position and orientation. Estimating the fish pose can therefore be divided into two sub-problems: position and orientation.

Position Estimate

In terms of position estimation, it is reasonable to mention another common sensor for UVMS, which is a stereo camera. Stereo cameras function based on the principle of stereoscopic vision, which mimics the depth perception capabilities of human eyes. These cameras use two lenses positioned at slightly different angles and positions such that depth can be estimated. Figure 16 illustrates how two lenses can be used to get depth information. Since the system at hand is only equipped with monocular vision there is no direct information about depth, but it does include two cameras. Therefore, the same method could be applied with the knowledge of the position of each camera.

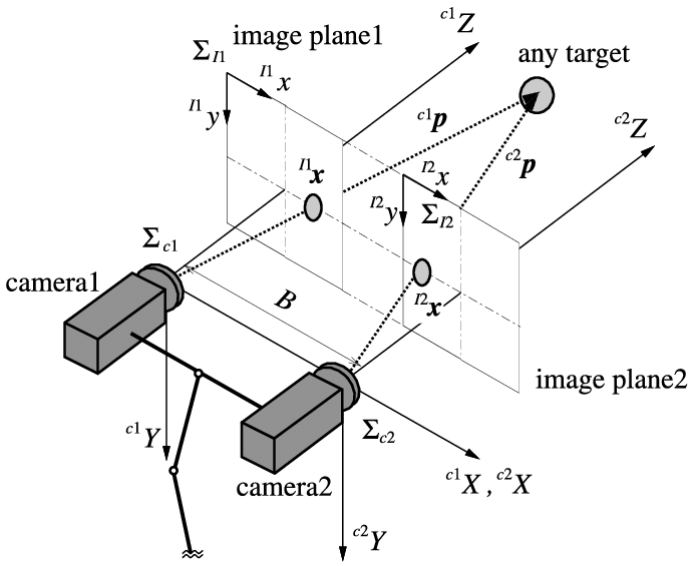


Figure 16: Illustration of how stereo cameras can be used to get depth information [21].

Regardless, methods utilizing monocular vision are useful in the pursuit of low-cost systems. Therefore, Haugaløkken et al. [18] experimented with monocular vision-based gripping of objects. By comparing the known spatial features of the object with its image, they were able to estimate its position in Cartesian space.

Orientation Estimate

In Ming En Koh et al. [19] they used the bounding box of the fish in addition to the bounding box of the head to estimate the orientation of the fish as seen in Fig. 17.

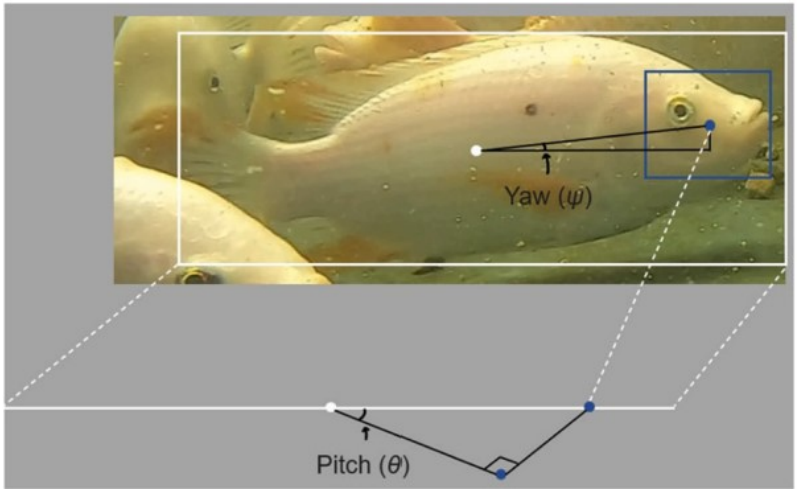


Figure 17: Estimating fish pose with bounding box of head and body [19].

The relative position of the head to the fish body is then used to estimate the pose of the fish using trigonometric identities [19]

$$\text{pitch} = \arccos\left(\frac{\text{fish}_{x_{\text{avg}}} - \text{head}_{x_{\text{avg}}}}{\text{fish}_{x_{\text{max}}} - \text{fish}_{x_{\text{avg}}} - (\text{head}_{x_{\text{max}}} - \text{head}_{x_{\text{avg}}})}\right) \tag{17a}$$

$$\text{yaw} = \arctan\left(\frac{\text{head}_{y_{\text{avg}}} - \text{fish}_{y_{\text{avg}}}}{\text{head}_{x_{\text{avg}}} - \text{fish}_{x_{\text{avg}}}}\right). \tag{17b}$$

4 Vehicle-Manipulator System Dynamics

Integrating an AUV with a robotic arm transforms it into a vehicle-manipulator system or, more specifically, an Underwater Vehicle-Manipulator System (UVMS). This combination results in a time-variant system due to changes in inertia, mass distribution, and buoyancy caused by the movement of the robotic arm. Furthermore, this integration causes interaction dynamics, where the movement of the robotic arm and the AUV can disrupt each other's performance. Additionally, hydrodynamic effects contribute further to the system's complexity, affecting its operational efficiency and control strategies. This section will introduce the dynamic modeling of an AUV, building on the theory outlined by Fossen [22]. It will then explore the modeling of a vehicle-manipulator system. Finally, the section will detail the formulation of the manipulator Jacobian for the UVMS.

4.1 Dynamic Modelling

A simulation of the system will be developed, which requires dynamic modeling for accurate representation. This section will cover the various components that could be incorporated into the dynamics to improve realism, emphasizing the mathematical aspects pertinent to this domain. While simplifying the dynamics may be acceptable for manageability, it is crucial to identify and consider all factors that could influence the system's behavior to design and validate the control system effectively.

4.1.1 AUV

Let's first look at the dynamics of the AUV. From Fossen's handbook [22], the equations of motion can be expressed as,

$$\dot{\boldsymbol{\eta}} = \mathbf{J}_k(\boldsymbol{\eta})(\boldsymbol{\nu}_r + \boldsymbol{\nu}_c) \tag{18a}$$

$$\mathbf{M}\dot{\boldsymbol{\nu}}_r + \mathbf{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \mathbf{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau}, \tag{18b}$$

where

$\mathbf{M}$	-	is the system inertia matrix including added mass,
$\mathbf{C}(\boldsymbol{\nu}_r)$	-	is the Coriolis–centripetal matrix also including added mass,
$\mathbf{D}(\boldsymbol{\nu}_r)$	-	is the damping matrix,
$\mathbf{g}(\boldsymbol{\eta})$	-	is the vector of gravitational/buoyancy forces and moments,
$\boldsymbol{\tau}$	-	represents the wind, wave, and propulsion forces,
$\boldsymbol{\eta}$	-	is the position and orientation state vector $\begin{bmatrix} x, y, z, \phi, \theta, \psi \end{bmatrix}^T$.
$\boldsymbol{\nu}_r$	-	is the relative linear and angular velocity $\boldsymbol{\nu}_r = \boldsymbol{\nu} - \boldsymbol{\nu}_c$,
$\boldsymbol{\nu}$	-	is the linear and angular velocity state vector $\begin{bmatrix} u, v, w, p, q, r \end{bmatrix}^T$,
$\boldsymbol{\nu}_c$	-	is the linear and angular velocity of the current.

The following subsections will delve into the specifics of constructing these matrices, illuminating their physical significance and their roles in the AUV's dynamic model.

System Inertia Matrix, $\boldsymbol{M}$

The system inertia matrix, denoted as $\boldsymbol{M}$, encapsulates the mass and inertia characteristics of the AUV, integrating both the rigid-body inertia and the hydrodynamic added mass effects. The added mass effect is particularly significant, as it represents the inertia added to the system due to the displacement of water when the AUV moves. This can be formally expressed as follows,

$$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A. \tag{19}$$

Here, $\boldsymbol{M}_{RB}$ signifies the rigid-body inertia matrix, constructed based on the principles of symmetry and physical considerations of mass distribution as outlined in Fossen's handbook [22]

$$\boldsymbol{M}_{RB} = \begin{bmatrix} m\mathbf{I}_3 & -m\mathbf{S}(\boldsymbol{r}_{bg}^b) \\ m\mathbf{S}(\boldsymbol{r}_{bg}^b) & \mathbf{I}_b^b \end{bmatrix}, \tag{20}$$

where m is the mass, I is the identity matrix of size 3×3, r_{bg}^b is the vector representing the position of the COM in the body frame, and $\mathbf{I}_b^b$ denotes the body-frame inertia tensor, capturing the AUV's resistance to angular acceleration, detailed as

$$\mathbf{I}_b^b = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}. \tag{21}$$

The added mass matrix, $\boldsymbol{M}_A$, incorporates hydrodynamic effects due to the vehicle's interaction with surrounding water and is initially expressed through detailed hydrodynamic coefficients found through system identification, resulting in

$$\boldsymbol{M}_A = -\begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} \end{bmatrix}. \tag{22}$$

For practical purposes, and in situations where hydrodynamic interactions are primarily linear, a simplification can be made by using only the diagonal terms of the matrix such that

$$\boldsymbol{M}_A = -\mathrm{diag}(X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}, K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}). \tag{23}$$

Coriolis-Centripetal Matrix, $\boldsymbol{C}(\boldsymbol{\nu}_r)$

Following the definition of the system inertia matrix, the next step involves formulating the Coriolis-centripetal matrix, $\boldsymbol{C}(\boldsymbol{\nu}_r)$. This matrix plays a crucial role in the dynamic model by accounting for the effects arising from the AUV's rotational and translational motion. Specifically, it represents the coupling between different modes of motion, which is essential for accurate modeling of the AUV's behavior in a fluid environment.

The matrix $\boldsymbol{C}(\boldsymbol{\nu}_r)$ is directly derived from the mass matrix $\boldsymbol{M}$, reflecting how changes in the vehicle's velocity affect the distribution of forces and moments acting on the vehicle. It is calculated using the following expression,

$$\boldsymbol{C}(\boldsymbol{\nu}_r) = \begin{bmatrix} \boldsymbol{O}_3 & -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{v}_{nb}^b + \boldsymbol{M}_{12}\boldsymbol{\omega}_{nb}^b) \\ -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{v}_{nb}^b + \boldsymbol{M}_{12}\boldsymbol{\omega}_{nb}^b) & -\boldsymbol{S}(\boldsymbol{M}_{21}\boldsymbol{v}_{nb}^b + \boldsymbol{M}_{22}\boldsymbol{\omega}_{nb}^b) \end{bmatrix}, \tag{24}$$

where $\boldsymbol{O}_3$ represents a zero matrix of size 3×3, $\boldsymbol{M}_{nn}$ are submatrices from the system inertia matrix, $\boldsymbol{M}$, and $\boldsymbol{v}_{nb}^b$ and $\boldsymbol{\omega}_{nb}^b$ is the linear and angular velocity vectors of the AUV in body frame, respectively.

Damping Matrix, $\boldsymbol{D}(\boldsymbol{\nu}_r)$

In underwater vehicle dynamics, the concept of damping is integral to capturing the resistance that the vehicle experiences as it moves through water. The damping forces and moments acting on an underwater vehicle can be highly nonlinear and coupled, particularly when operating at high speeds or in turbulent conditions. This nonlinearity is reflected in the complexity and cross-coupling terms of the damping matrix [22].

However, the damping matrix can be considerably simplified under slow-motion conditions or when the vehicle's movements are not coupled. For these scenarios, the damping forces can be approximated as being predominantly linear or quadratic with respect to the vehicle's velocities. This simplification leads to a damping matrix that can be structured in a diagonal form, significantly reducing computational complexity while providing a reasonable approximation of the vehicle's dynamics. Leading to

$$\boldsymbol{D}(\boldsymbol{\nu}_r) = \boldsymbol{D}_{\text{linear}} + \boldsymbol{D}_{\text{qudratic}}, \tag{25}$$

where

$$\boldsymbol{D}_{\text{linear}} = -\text{diag}([X_u, Y_v, Z_w, K_p, M_q, N_r]) \tag{26a}$$

$$\boldsymbol{D}_{\text{quadratic}} = -\text{diag}([X_{|u|u}|u|, Y_{|v|v}|v|, Z_{|w|w}|w|, K_{|p|p}|p|, M_{|q|q}|q|, N_{|r|r}|r|]), \tag{26b}$$

u, v, w, p, q, and r are the linear and angular velocities across the 6 DOF, respectively. The coefficients $X_u, Y_v, Z_w, K_p, M_q, N_r$ and $X_{|u|u}, Y_{|v|v}, Z_{|w|w}, K_{|p|p}, M_{|q|q}, N_{|r|r}$ are constant parameters determined through system identification processes. These constants represent the linear and quadratic damping factors in each respective DOF, characterizing the resistances the vehicle encounters as it moves through the water.

4.1.2 Vehicle-Manipulator

The equations previously discussed do not encompass the dynamics introduced by integrating a robotic arm. Therefore, the equations of motion in (18) must be appropriately modified to encapsulate the combined dynamics of the AUV and the manipulator. As proposed by Chang et al. [23], the extended equations of motion can be expressed as

$$\dot{\boldsymbol{\eta}}_{ee} = \boldsymbol{J}_k(\boldsymbol{R}_B^I, \boldsymbol{q}_m)\boldsymbol{\zeta} \tag{27a}$$

$$\boldsymbol{M}(\boldsymbol{\eta}_v, \boldsymbol{q}_m)\dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\boldsymbol{\eta}_v, \boldsymbol{q}_m, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{D}(\boldsymbol{\eta}_v, \boldsymbol{q}_m, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{G}(\boldsymbol{\eta}_v, \boldsymbol{q}_m) = \boldsymbol{\tau}_e + \boldsymbol{\tau}_c, \tag{27b}$$

where $\boldsymbol{\zeta}$ denotes the comprehensive state vector that includes the states of both the AUV and the manipulator arm. It is imperative to note that the system matrices are functions of the joint angles $\boldsymbol{q}_m$ of the robotic arm, indicating that the dynamic properties of the system will vary with changes in the manipulator's configuration.

Including the robotic arm introduces additional layers of complexity to the dynamic modeling. These complexities arise from the arm's movements and interactions with the environment, subsequently influencing the AUV's behavior. This interaction demands an examination of the forces and moments exerted at the connection point between the AUV and the robotic arm. The vehicle-manipulator system's dynamic equations are thus significantly more intricate, as they encapsulate the coupled dynamics of the underwater vehicle and the multi-linked manipulator.

Constructing these equations involves accounting for the kinematic and dynamic interplay between the AUV and the manipulator arm. This includes considering the effect of the manipulator's configuration on the AUV's inertia, buoyancy, and hydrodynamic characteristics. Due to the mathematical complexity and the computational demands of deriving and solving these comprehensive dynamic equations, this analysis poses significant challenges.

In light of these complexities, while a theoretical framework for these dynamics is critical for understanding the vehicle-manipulator system, practical implementation, and simulation have been approached differently. For this research, a physics engine, specifically Simscape in MATLAB, is employed to simulate the vehicle-manipulator dynamics, with the manipulator-induced forces and moments considered as disturbances to the AUV system. This pragmatic approach allows for examining the system's response to manipulator movements without directly solving the complete set of dynamic equations. A more detailed discussion on the choice of simulation tools and strategies will be presented in Section 6.

4.2 Vehicle-Manipulator Jacobian

Section 2.3.1 presented the construction of the manipulator Jacobian, which is defined with respect to the manipulator's base frame. However, this thesis addresses the dynamics of a vehicle-manipulator system, necessitating the formulation of a manipulator Jacobian matrix that represents the system in its entirety with respect to the world frame.

First, let's examine the Jacobian of the AUV, encapsulating its contribution to end-effector motion. The linear motion of the end-effector, relative to the world frame, mirrors the linear movement of the AUV. Therefore, by examining the AUV's frame with respect to the world frame, the linear components of the Jacobian matrix are derived. The orientation of the AUV in the world frame is represented as

$$\boldsymbol{x}_{\text{auv}} = \boldsymbol{R}^{\text{auv}}_{\text{world}} \cdot \vec{\boldsymbol{x}} \tag{28a}$$
$$\boldsymbol{y}_{\text{auv}} = \boldsymbol{R}^{\text{auv}}_{\text{world}} \cdot \vec{\boldsymbol{y}} \tag{28b}$$
$$\boldsymbol{z}_{\text{auv}} = \boldsymbol{R}^{\text{auv}}_{\text{world}} \cdot \vec{\boldsymbol{z}}, \tag{28c}$$

where $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$, and $\vec{\boldsymbol{z}}$ are the unit vectors $[0,1,0]^T$, $[0,0,1]^T$, and $[1,0,0]^T$ respectively. Then the linear part of the Jacobian becomes

$$\boldsymbol{J}_{v,x} = \boldsymbol{x}_{\text{auv}} \tag{29a}$$
$$\boldsymbol{J}_{v,y} = \boldsymbol{y}_{\text{auv}} \tag{29b}$$
$$\boldsymbol{J}_{v,z} = \boldsymbol{z}_{\text{auv}}. \tag{29c}$$

The impact on the end-effector's linear velocity because of the AUV's angular velocity arises from the spatial displacement of the end-effector from the AUV's center of rotation, giving the Jacobian components

$$\boldsymbol{J}_{v,\text{roll}} = \boldsymbol{x}_{\text{auv}} \times \boldsymbol{p}^{\text{auv}}_{\text{ee}} \tag{30a}$$
$$\boldsymbol{J}_{v,\text{pitch}} = \boldsymbol{y}_{\text{auv}} \times \boldsymbol{p}^{\text{auv}}_{\text{ee}} \tag{30b}$$
$$\boldsymbol{J}_{v,\text{yaw}} = \boldsymbol{z}_{\text{auv}} \times \boldsymbol{p}^{\text{auv}}_{\text{ee}}, \tag{30c}$$

where $\boldsymbol{p}^{\text{auv}}_{\text{ee}}$ denotes the position vector of the end-effector relative to the AUV's center of rotation.

Furthermore, the AUV's angular velocities directly affect the end-effector's orientation. Depending on the orientation of the end-effector relative to the AUV frame, the angular velocity contribution can be expressed as

$$\boldsymbol{J}_{\omega,\text{roll}} = \boldsymbol{R}^{\text{auv}}_{\text{ee}} \cdot \boldsymbol{x}_{\text{auv}} \tag{31a}$$
$$\boldsymbol{J}_{\omega,\text{pitch}} = \boldsymbol{R}^{\text{auv}}_{\text{ee}} \cdot \boldsymbol{y}_{\text{auv}} \tag{31b}$$
$$\boldsymbol{J}_{\omega,\text{yaw}} = \boldsymbol{R}^{\text{auv}}_{\text{ee}} \cdot \boldsymbol{z}_{\text{auv}}. \tag{31c}$$

Integrating these components yields the complete Jacobian for the AUV,

$$\boldsymbol{J}(\boldsymbol{\eta}(t)) = \begin{bmatrix} [\boldsymbol{J}_{v,x}, \boldsymbol{J}_{v,y}, \boldsymbol{J}_{v,z}] & [\boldsymbol{J}_{v,\text{roll}}, \boldsymbol{J}_{v,\text{pitch}}, \boldsymbol{J}_{v,\text{yaw}}] \\ \boldsymbol{0}_{3\times 3} & [\boldsymbol{J}_{\omega,\text{roll}}, \boldsymbol{J}_{\omega,\text{pitch}}, \boldsymbol{J}_{\omega,\text{yaw}}] \end{bmatrix}, \tag{32}$$

which encapsulates both linear and angular contributions to the end-effector's motion.

Finally, the manipulator Jacobian constructed in Section 2.3.1 needs to be rotated from the base frame to the world frame to get a complete Jacobian of the system as

$$\boldsymbol{J}(\boldsymbol{\eta}, \boldsymbol{q}) = \begin{bmatrix} \boldsymbol{J}(\boldsymbol{\eta}(t)) & \boldsymbol{J}(\boldsymbol{q}(t)) \end{bmatrix}. \tag{33}$$

5 Control System for UVMS

A robotic system is defined as kinematically redundant if it possesses more DOFs than what is strictly necessary to accomplish a given task [24]. This redundancy allows for multiple configurations to achieve the same end goal, introducing complexity and versatility in control approaches. The integration of an AUV with five DOFs and a robot manipulator with four DOFs exemplifies such redundancy, as a total of six DOFs is sufficient for orienting the end effector in any desired position and orientation. However, the excess DOFs facilitate a broader range of movement and potential operational strategies, while demanding more from the control system in terms of sophistication and adaptability.

Control approaches for such systems are broadly categorized into kinematic control and dynamic control, where kinematic control is used to generate desired trajectories and dynamic control is used to make the system follow these trajectories. This section will briefly explore the different control methods for UVMS within these two main categories and additionally mention the possibility of separated control. It will assess their respective advantages and challenges to argue for an appropriate control strategy for the task at hand. The book *Underwater Robots* by Gianluca Antonelli [24] is the main source for this argumentation and presented equations.

5.1 Kinematic Control

Kinematic control exposes the relationship between the velocities of a system's components and the motion of its end-effector. In manipulation tasks, the objective is typically specified as a trajectory for the end-effector's pose. However, control inputs are applied through actuators that influence the system's velocities. It is essential, then, to determine the system velocities that will produce the intended motion of the end-effector. This section introduces how a task is defined and its corresponding Jacobian. It also addresses one of the key challenges in implementing inverse kinematics, the emergence of singularities within the Jacobian matrix. To navigate this challenge, the use of a pseudo-inverse of the Jacobian is explored. Lastly, the section introduces the task-priority framework, a strategic approach to managing multiple tasks.

5.1.1 Defining Tasks

Antonelli provides a list of possible tasks for a UVMS [24]:

- End-effector position norm
- End-effector obstacle avoidance
- End-effector position
- End-effector orientation
- End-effector configuration
- End-effector field of view
- End-effector relative field of view
- Mechanical joint-limit
- Robot manipulability
- Robot nominal configuration
- Vehicle orientation
- Vehicle yaw
- Vehicle roll-pitch
- Vehicle-fixed sensor field of view
- Vehicle-fixed relative field of view
- Vehicle position norm
- Vehicle obstacle avoidance

The potential tasks a system can execute are bounded only by the constraints of its configuration and the creativity with which one approaches task design. Given a particular system setup and an objective, the diversity of attainable tasks is limited solely by the ingenuity applied in defining them. The task, χ, is defined as

$$\boldsymbol{\sigma}_\chi = \boldsymbol{\sigma}_\chi(\boldsymbol{\eta}, \boldsymbol{q}) \tag{34}$$

where $\boldsymbol{J}_\chi$ is the corresponding Jacobian relating its time derivative to the system velocity [24]

$$\dot{\boldsymbol{\sigma}}_\chi = \boldsymbol{J}_\chi(\boldsymbol{\eta}, \boldsymbol{q})\boldsymbol{\zeta} \tag{35}$$

5.1.2 Psuedo-Inverse of the Jacobian

As mentioned, the inverse of the Jacobian may lead to difficulties in terms of singularities. Therefore, there is the option of using the pseudo-inverse to calculate the desired joint velocities from a desired end-effector velocity [24]

$$\boldsymbol{\zeta}_r = \boldsymbol{J}_\chi^\dagger \dot{\boldsymbol{\sigma}}_\chi, \tag{36}$$

where $\boldsymbol{J}_\chi^\dagger$ is the Moore-Penrose psuedo-inverse calculated as,

$$\boldsymbol{J}_\chi^\dagger = \boldsymbol{J}_\chi^T \left(\boldsymbol{J}_\chi \boldsymbol{J}_\chi^T\right)^{-1}. \tag{37}$$

One can also calculate a weighted pseudo-inverse as

$$\boldsymbol{J}_{\chi,W}^\dagger = \boldsymbol{W}^{-1}\boldsymbol{J}_\chi^T \left(\boldsymbol{J}_\chi \boldsymbol{W}^{-1} \boldsymbol{J}_\chi^T\right)^{-1}, \tag{38}$$

where $\boldsymbol{W}$ is a diagonal matrix with weights penalizing effort in the different joints.

5.1.3 Transpose of the Jacobian

A simple approach to get desired end-effector velocities based on system velocities is using the transpose of the Jacobian in combination with a gain $\boldsymbol{K}_\chi$ [24], resulting in

$$\boldsymbol{\zeta}_r = \boldsymbol{J}^T \boldsymbol{K}_\chi \tilde{\boldsymbol{\sigma}}_\chi \tag{39}$$

This method does not directly compute the end-effector velocities but serves as a proportional control mechanism that uses the error, $\tilde{\boldsymbol{\sigma}}$, to adjust the velocities.

5.1.4 Task-Priority

In the context of manipulation tasks, the ability to effectively manage multiple subtasks is crucial, and this is where the concept of task priority becomes essential. Additionally, in kinematically redundant systems, such as the UVMS addressed in this study, the capacity

to execute several tasks concurrently exists. This is facilitated by utilizing the surplus DOFs inherent in the system. Such redundancy enables the simultaneous fulfillment of a primary task while also engaging in additional tasks, thereby optimizing the overall functionality and efficiency of the UVMS.

Lets consider the two tasks $\boldsymbol{\sigma}_a$ and $\boldsymbol{\sigma}_b$ with its associated Jacobians $\boldsymbol{J}_a$ and $\boldsymbol{J}_b$. Then, the task-priority framework finds the reference velocities

$$\boldsymbol{\zeta}_r = \boldsymbol{J}_a^\dagger \dot{\boldsymbol{\sigma}}_a + \boldsymbol{N}_a \boldsymbol{J}_b^\dagger \dot{\boldsymbol{\sigma}}_b, \tag{40}$$

where $\boldsymbol{N}_a$ projects a generic joint velocity vector in the null space of the Jacobian matrix $\boldsymbol{J}_a$ [24] expressed as

$$\boldsymbol{N}_a = \boldsymbol{I}_n - \boldsymbol{J}_a^\dagger \boldsymbol{J}_a. \tag{41}$$

This corresponds to generating a motion that does not affect the task space $\boldsymbol{\sigma}_a$. Extending this method to accommodate multiple tasks is feasible, yet it requires careful consideration to avoid conflicts between task priorities and dynamics. The following equation illustrates how to expand this model to multiple tasks, as detailed by Antonelli [24],

$$\boldsymbol{\zeta}_r = \boldsymbol{J}_a^\dagger \dot{\boldsymbol{\sigma}}_a + \boldsymbol{N}_a \boldsymbol{J}_b^\dagger \dot{\boldsymbol{\sigma}}_b + \boldsymbol{N}_{ab} \boldsymbol{J}_c^\dagger \dot{\boldsymbol{\sigma}}_c \tag{42}$$

This formulation integrates several task spaces, where $\boldsymbol{J}_a^\dagger$, $\boldsymbol{J}_b^\dagger$, and $\boldsymbol{J}_c^\dagger$ represent the pseudoinverses of the Jacobians for respective tasks, and $\boldsymbol{N}_a$, $\boldsymbol{N}_{ab}$ are the null space projectors that ensure each subsequent task does not interfere with the execution of the preceding ones.

5.1.5 Actuator Singularity Avoidance

Singularity is a well-known issue in robotic manipulators, making it essential to explore options for singularity avoidance. This section will examine singularity avoidance within a task priority framework, which has been addressed in several studies. Specifically, the Master's theses by Stene [25] and Basso [26] employ this method, utilizing the manipulability index introduced in Section 2.4.1 to define the task

$$\sigma_b := \det(\boldsymbol{B}(\boldsymbol{q})\boldsymbol{B}(\boldsymbol{q})^T), \tag{43}$$

and computing the task Jacobian as

$$\boldsymbol{J}_b = \begin{bmatrix} \frac{\partial \sigma_b}{\partial q_1} & \frac{\partial \sigma_b}{\partial q_2} & \frac{\partial \sigma_b}{\partial q_3} & \cdots & \frac{\partial \sigma_b}{\partial q_i} \end{bmatrix}, \tag{44}$$

where

$$\frac{\partial \sigma_b}{\partial q_i} = 2\sigma_b \operatorname{Tr}(\frac{\partial \boldsymbol{B}}{\partial q_i} \boldsymbol{B}^\dagger). \tag{45}$$

This task is further applied as a set-based singularity avoidance task. As proposed by Moe [27], "several goals may be described as set-based tasks, which are tasks that aim for a desired range of values rather than a precise singular value." This approach is particularly relevant to the singularity avoidance task from equation (43), where maintaining the manipulability index within a specified range is deemed appropriate. By defining a feasible set

$$D = \{\sigma \in \mathbb{R} \mid \sigma_{\min} \leq \sigma \leq \sigma_{\max}\} \tag{46}$$

the singularity avoidance task should be active when $\sigma \notin D$. Thus, when the manipulability index is within the feasible set, D, it should not influence the system. Sverdrup-Thygeson et al. [28] applied this method on an underwater swimming manipulator, demonstrating its relevance to the current task at hand.

5.1.6 Secundary Task with Full Rank Main Task

The task-priority framework requires the Jacobian to lose rank for the projection into the null space to enable switching between tasks. Marey and Chaumette [10] proposed an alternative approach for transitioning to a secondary task when the Jacobian maintains full rank in the primary task, utilizing a sigmoid function

$$\Lambda = \frac{1}{1 + \exp(12\frac{\|\tilde{\boldsymbol{\sigma}}\| - e_o}{e_1 - e_0} + 6)}, \tag{47}$$

using the norm of the task error, $\|\tilde{\boldsymbol{\sigma}}\|$, such that when the task error is small, the function converges to 0. e_0 and e_1 is used to tune the response of the function as shown in Figure 18.

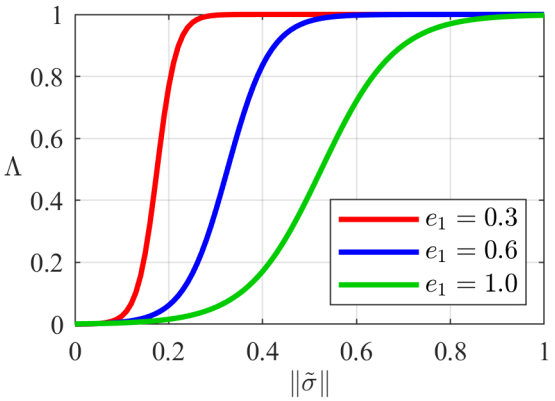


Figure 18: Sigmoid function with $e_0 = 0.05$ and different e_1.

To effectively transition between tasks, the parameter Λ is strategically incorporated into the control scheme from equation (40) and replaces the null space projection $\boldsymbol{N}_a$ as follows,

$$\boldsymbol{\zeta}_r = \Lambda \boldsymbol{J}_a^{\dagger} \dot{\boldsymbol{\sigma}}_a + (1 - \Lambda) \boldsymbol{J}_b^{\dagger} \dot{\boldsymbol{\sigma}}_b. \tag{48}$$

5.2 Separated Control

Robust and established solutions exist for controlling an AUV and a robotic manipulator independently. Therefore, one potential control strategy could involve treating the systems separately and adopting a well-established control approach for each. Typically, one method involves maintaining the AUV's position and orientation while the robotic arm carries out manipulation tasks. Alternatively, it is conceivable to not actively control the AUV during manipulation tasks, allowing it to serve as a floating base for the robotic manipulator. Chang et al. [23] explored various scenarios, initially implementing separated control with only positional control on the AUV, followed by experiments with both positional and attitude control. They concluded that maintaining the vehicle's position and orientation is not feasible due to the dynamic coupling between the manipulator and the vehicle, which adversely affects the precision of the manipulator's end effector and, consequently, undermines the accuracy required for underwater operations.

Using the AUV as a floating base is contingent upon a significant inertia ratio between the vehicle and the manipulator. Chang et al. [23] reported a case where the vehicle weighed approximately 5500 kg and the arm 65 kg, resulting in a ratio of approximately 84.6. While seemingly large, the underwater environment can mitigate this advantage due to the easier movement of submerged objects, meaning that coupling effects can still significantly impact performance. In the context of the BlueROV2 and Reach Alpha 5 with their approximate weight of 12 kg and 1.2 kg, the inertia ratio is considerably smaller, suggesting that a separated control approach may not be optimal for this specific configuration.

5.3 Dynamic Control

Dynamic control strategies fundamentally rely on an accurate dynamic model of the system, as they directly incorporate the system's dynamic equations into the control law. As elaborated in Section 4, the derivation of dynamic equations reveals that kinematically redundant systems result in a complex equation set. Despite this complexity, the advent of sophisticated computational resources and advanced system identification techniques has made dynamic control increasingly feasible and potent. The control strategies discussed are explored in Gianluca Antonelli's book *Underwater Robots* [24], but revisited to provide a succinct overview of different dynamic control approaches relevant to the task at hand.

Feedforward Decoupling Control

In the complex domain of UVMS, the interaction forces between the vehicle and manipulator present significant control challenges. An innovative approach to mitigate these challenges involves using Feedforward Decoupling Control. As proposed by McLain et al. [29], this strategy entails estimating forces and moments generated by the manipulator's movements and compensating for these in the vehicle's control.

The dynamic model for the vehicle part of the UVMS can be represented as

$$\boldsymbol{M}_v\dot{\boldsymbol{\nu}} + \boldsymbol{C}_v(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}_{\mathrm{RB}}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}_{\mathrm{RB}}(\boldsymbol{R}_B^I) = \boldsymbol{\tau}_v - \boldsymbol{\tau}_m(\boldsymbol{R}_B^I, \boldsymbol{q}, \boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}}) \tag{49}$$

where $\boldsymbol{\tau}_m$ signifies the forces and moments induced by the manipulator's movements. By accurately estimating τ_m, the control law can be refined to

$$\boldsymbol{\tau}_v = \boldsymbol{\tau}_{\text{control}} + \hat{\boldsymbol{\tau}}_m(\boldsymbol{R}_B^I, \boldsymbol{q}, \boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}}) \tag{50}$$

This method relies on the precision of the model used to predict hydrodynamic interaction forces. When these forces are accurately predicted and fed into the vehicle controller, the UVMS's station-keeping performance is significantly improved, as evidenced by McLain et al. [29].

Feedback Linearization

Using the non-linear dynamic model for a UVMS and linearizing around the working point allows for a detailed control approach. As shown in the research by Tarn et al. [30], the control law

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{\zeta}}_a + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{D}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{R}_B^I) \tag{51}$$

provides a means to manage the system's complex interactions. The model, which accounts for hydrodynamic forces, has been tested on a vehicle with two manipulators. The results from these tests indicated that the control scheme effectively coordinates the vehicle and manipulator movements, ensuring stable tracking and maneuvering.

Nonlinear Control for UVMSs with Composite Dynamics

The vehicle and manipulator parts of a UVMS have different bandwidth characteristics, both in terms of dynamics and hardware. The dynamics differ because of the difference in form and size, resulting in different inertia and hydrodynamics. Regarding hardware, the actuators and sensors may differ in the response time and sampling frequency. Canudas de Wit et al. [31] addresses the challenge of composite dynamics by proposing a tailored non-linear control

$$\boldsymbol{\tau}_v^* = (\hat{\boldsymbol{M}}_v^* + \hat{\boldsymbol{M}}_{qq})(\ddot{\boldsymbol{\nu}}_d + k_{vv}\dot{\tilde{\boldsymbol{\nu}}} + k_{vp}\tilde{\boldsymbol{\nu}}) + \Delta\boldsymbol{\tau}_v^* \tag{52a}$$

$$\boldsymbol{\tau}_q = \hat{\boldsymbol{M}}_q(\ddot{\boldsymbol{q}}_d + k_{vq}\dot{\tilde{\boldsymbol{q}}} + k_{pq}\tilde{\boldsymbol{q}}) \tag{52b}$$

where the gains, k_{vv}, k_{vp}, k_{vq} and k_{pq} are chosen based on the systems bandwidth and $\Delta\boldsymbol{\tau}_v^*$ is an additional control action to compensate the decoupling forces which can be chosen in different ways [24].

Sliding Mode Control

Sliding mode control (SMC) is a robust control technique that effectively handles model uncertainties and external disturbances. This approach is particularly beneficial in environments with high variability and unpredictability, such as underwater dynamics. A key component of SMC is the definition of a sliding surface, which the control system aims to reach. This is mathematically represented by

$$\boldsymbol{s}(\boldsymbol{x}, t) = \dot{\tilde{\boldsymbol{x}}} + \boldsymbol{\Lambda}\tilde{\boldsymbol{x}} = 0 \tag{53}$$

where $\tilde{\boldsymbol{x}}$ denotes the deviation from the desired trajectory, and $\boldsymbol{\Lambda}$ is a positive definite matrix that defines the dynamics of the sliding surface. With the dynamic equation of a UVMS

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + D(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{R}_B^I) = \boldsymbol{B}\boldsymbol{u} \tag{54}$$

where $\boldsymbol{B}$ is the input matrix, mapping control inputs $\boldsymbol{u}$ to forces and torques. The control input $\boldsymbol{u}$ is derived as

$$\boldsymbol{u} = \boldsymbol{B}^{\dagger} \left[\boldsymbol{K}_D \boldsymbol{s} + \hat{\boldsymbol{g}}(\boldsymbol{q}, \boldsymbol{R}_B^I) + \boldsymbol{K}_S \text{sign}(\boldsymbol{s}) \right], \tag{55}$$

where $\boldsymbol{B}^{\dagger}$ is the pseudoinverse of $\boldsymbol{B}$, $\boldsymbol{K}_D$ is a gain matrix, $\hat{\boldsymbol{g}}(\boldsymbol{q}, \boldsymbol{R}_B^I)$ is the estimate of gravitational and buoyant forces, and $\boldsymbol{K}_S$ is a switching gain that enhances the system's robustness against disturbances and model uncertainties through the term sign($\boldsymbol{s}$), promoting quick convergence to the sliding surface. This configuration allows the UVMS to adapt quickly to changes in its environment, ensuring stability and accuracy of the control performance under varying underwater conditions. One notable drawback of SMC is chattering in the control inputs due to the switching occurring in the term sign($\boldsymbol{s}$) [24].

Adaptive Control

Adaptive control is essential for ensuring the robust performance of dynamic systems such as the UVMS under varying operational conditions. This control strategy dynamically adjusts control gains in response to system parameter changes, helping maintain optimal performance despite environmental disturbances and parameter uncertainties.

The adaptive control mechanism for the UVMS can be described by the following dynamic equation:

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{D}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{g}(\boldsymbol{q}, \boldsymbol{R}_B^I) = \boldsymbol{\Phi}(\boldsymbol{q}, \boldsymbol{R}_I^B, \boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}})\boldsymbol{\theta} = \boldsymbol{B}\boldsymbol{u}. \tag{56}$$

This equation models the forces and torques acting on the system, with $\boldsymbol{\Phi}(\boldsymbol{q}, \boldsymbol{R}_I^B, \boldsymbol{\zeta}, \dot{\boldsymbol{\zeta}})\boldsymbol{\theta}$ representing the parametric uncertainties that the control system must adapt to. The control law is formulated as

$$\boldsymbol{u} = \boldsymbol{B}^{\dagger} \left[\boldsymbol{K}_D \boldsymbol{s}' + \boldsymbol{\Phi}(\boldsymbol{q}, \boldsymbol{R}_I^B, \boldsymbol{\zeta}, \boldsymbol{\zeta}_r, \dot{\boldsymbol{\zeta}}_r)\hat{\boldsymbol{\theta}} \right], \tag{57}$$

where $\boldsymbol{u}$ represents the control inputs adjusted by the pseudoinverse of the input matrix $\boldsymbol{B}$, $\boldsymbol{K}_D$ is the gain matrix for the controller, and $\hat{\boldsymbol{\theta}}$ is the estimate of the uncertain parameters affecting the system dynamics. The parameters are updated in real time according to

$$\dot{\hat{\boldsymbol{\theta}}} = \boldsymbol{K}_{\theta}^{-1} \boldsymbol{\Phi}^T(\boldsymbol{q}, \boldsymbol{R}_I^B, \boldsymbol{\zeta}, \boldsymbol{\zeta}_r, \dot{\boldsymbol{\zeta}}_r)\boldsymbol{s}. \tag{58}$$

This equation continuously adjusts $\hat{\boldsymbol{\theta}}$, ensuring the control system adapts to the real environment by compensating for the identified discrepancies.

6 Simulation Implementation

This section begins by highlighting the significance of simulation in designing control systems for complex robotic systems. It then moves on to discuss the selection of the simulation tool used. Following this, the section details the implementation of the simulation, which builds upon the dynamics outlined in Section 4. This theory is also used to defend the design decisions.

6.1 Importance of Simulation

In the realm of robotics and control systems, simulation plays a pivotal role in the research and development process. It provides a safe, cost-effective, and efficient means to test and refine control approaches under a wide range of conditions. Particularly in the context of this study, which focuses on underwater vehicle-manipulator systems, simulation becomes important due to the complex nature of the underwater environment.

A well-crafted simulation offers several advantages. Firstly, it enables the iterative testing of control strategies in a controlled environment, allowing for the systematic identification and rectification of errors before real-world implementation. This aspect is crucial in avoiding costly and potentially dangerous mistakes during physical deployment. Secondly, visual simulation adds an invaluable dimension to the research. It facilitates a more intuitive understanding of the system's behavior and responses. Through visual feedback, researchers can identify anomalies and inefficiencies that are difficult to detect through traditional plots.

Another critical aspect of effective simulation is realism. The quality of a simulation significantly affects how valid and applicable the research results are. In the context of underwater systems, this involves incorporating environmental dynamics specific to underwater settings, such as hydrodynamic forces. While certain simplifications can be made when assuming slow movements, making the simulation more manageable, they must be balanced carefully with the need for accuracy. A simulation that closely mirrors real-world physics will provide more reliable data and insights, leading to better-informed decisions and designs. However, achieving high levels of realism requires careful consideration of the assumptions and approximations made during the simulation design process. While it is not feasible to replicate every minute detail of the underwater environment, key factors such as buoyancy, drag, and fluid-structure interactions should be integrated to ensure that the simulation reflects the critical dynamics that occur in underwater environments.

In summary, the simulation component of this research is designed to provide a robust platform for testing and validating the control approach for the vehicle-manipulator system. By combining rigorous testing capabilities with high-fidelity environmental modeling, the simulation aims to bridge the gap between theoretical research and practical application, ensuring that the developed control strategies are effective and reliable in real-world underwater conditions.

6.2 Simulation Tool

An integral aspect of this study was selecting an appropriate simulation tool that integrates a physics engine capable of rendering the dynamics of the vehicle-manipulator system more intuitively than the standard dynamic equations outlined in Section 4.1. Initially,

the Robot Operating System (ROS) and Gazebo were considered due to their extensive flexibility and efficient communication protocols, which facilitate seamless integration with physical systems. These features are particularly beneficial for real-time applications and advanced robotic simulations. However, despite their advantages, the steep learning curve and time-intensive nature of mastering these tools posed significant challenges, especially given the limited prior experience with them.

Therefore, the decision was made to utilize MATLAB and Simulink for this project. These tools were chosen for several reasons: familiarity with the software, extensive support materials, and the availability of specific toolboxes tailored to robotic applications. Notably, the Simscape toolbox within MATLAB provides a robust platform for simulating the vehicle-manipulator system. It enables the modeling and simulation of physical systems across multiple domains, such as mechanical, electrical, hydraulic, and thermal systems within the Simulink environment. It allows users to represent physical components using a physical network approach rather than traditional signal-based modeling, making it easier to model real-world physical systems. Simscape provides a set of foundational libraries for various physical domains, and users can extend these libraries with custom components. The models in Simscape directly represent physical components, simplifying the process of translating engineering concepts into a simulation model. Additionally, it provides a visualization tool, Mechanics Explorer, for viewing and analyzing the simulation results of the physical model. It provides a 3D environment where users can interactively explore the motion and behavior of the mechanical models during and after simulation runs. Mechanics Explorer enables users to visualize the geometry, motion, and forces acting on components within a mechanical system, making it easier to understand the system's dynamics and to identify potential issues or areas for optimization. Figure 19 shows the visualization of the vehicle-manipulator system used in this thesis.

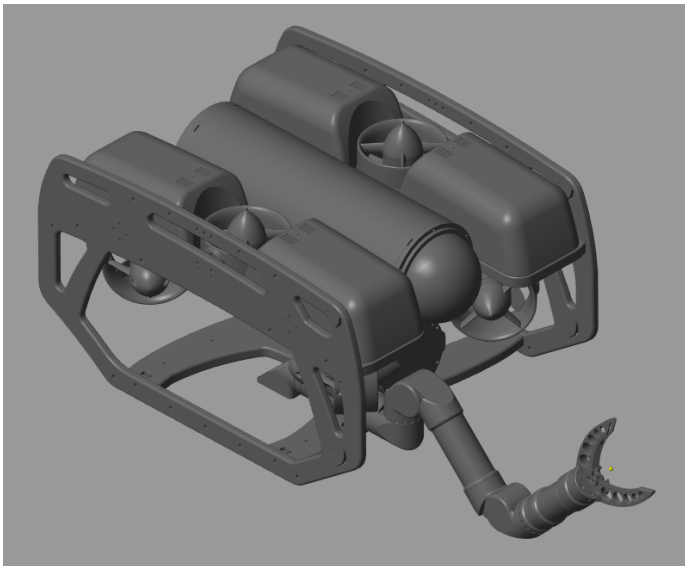


Figure 19: Mechanics Explorer visualization of BlueROV2 and Reach Alpha 5.

6.3 System Parameters

For enhanced realism, parameters from system identification should be employed in the simulator. While parameters specifically for the BlueROV2 are not available, system identification for the BlueROV1, a very similar vessel design, was performed by Sandøy

in 2016 [32]. Furthermore, the BlueROV2 Heavy has been simulated by Chu-Jou Wu [33], with certain parameters adjusted based on specifications provided by Blue Robotics, see the datasheet in Appendix B. Table 2 summarizes all parameters used for the dynamics of the AUV.

Table 2: Rigid body dynamics, added mass and damping parameters for the AUV.

Parameter	Value	Unit
Rigid body dynamics		
m	11.5	kg
W	112.8	N
B	114.8	N
r_b	$[0, 0, 0]^T$	m
r_g	$[0, 0, 0.02]^T$	m
I_x	0.16	kgm^2
I_y	0.16	kgm^2
I_z	0.16	kgm^2
Added mass		
$X_{\dot{u}}$	-5.5	kg
$Y_{\dot{v}}$	-12.7	kg
$Z_{\dot{w}}$	-14.57	kg
$K_{\dot{p}}$	-0.12	kgm^2/rad
$M_{\dot{q}}$	-0.12	kgm^2/rad
$N_{\dot{r}}$	-0.12	kgm^2/rad

Parameter	Value	Unit
Linear damping		
X_u	-4.03	Ns/m
Y_v	-6.22	Ns/m
Z_w	-5.18	Ns/m
K_p	-0.07	Ns/rad
M_q	-0.07	Ns/rad
N_r	-0.07	Ns/rad
Quadratic damping		
$X_{u\|u\|}$	-18.18	Ns^2/m^2
$Y_{v\|v\|}$	-21.66	Ns^2/m^2
$Z_{w\|w\|}$	-36.99	Ns^2/m^2
$K_{p\|p\|}$	-1.55	Ns^2/rad^2
$M_{q\|q\|}$	-1.55	Ns^2/rad^2
$N_{r\|r\|}$	-1.55	Ns^2/rad^2

Reach Robotics provided dynamic parameters for Reach Alpha 5 through their scientific paper *Kinematic and Dynamic Properties*, see Appendix A. However, the documentation does not include dynamic properties for the actuators and joints. To minimize oscillations during simulation, friction and damping were introduced into the joints. These modifications were selected to achieve a realistic response from the manipulator's behavior.

6.4 Simulation Structure

This section will explain the design choices and structure of the created simulation.

6.4.1 AUV

The AUV was simulated by employing the dynamic equations (18) from Section 4.1, utilizing parameters listed in Table 2. This was achieved by using a MATLAB function block for the equations and integrating the rate of change output, resulting in the scheme in Figure 20.

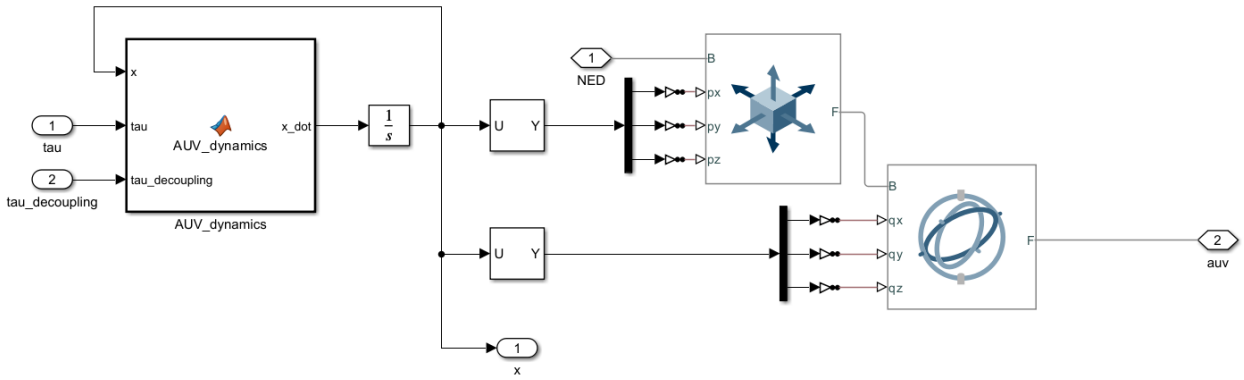


Figure 20: Dynamics of the AUV in Simulink.

The states were connected to a *Cartesian Joint* and *Gimbal Joint* with no resistance or dynamics, thus moving the reference frame of the AUV in 6 DOF. This ensures a smooth transition of the AUV dynamics to the Simscape environment. The AUV frame was then connected to a *Weld Joint*, which can be used to sense forces acting on the frame, making it possible to extract the decoupling forces from the robot arm and feed it to the AUV dynamics as a disturbance, see Figure 21.

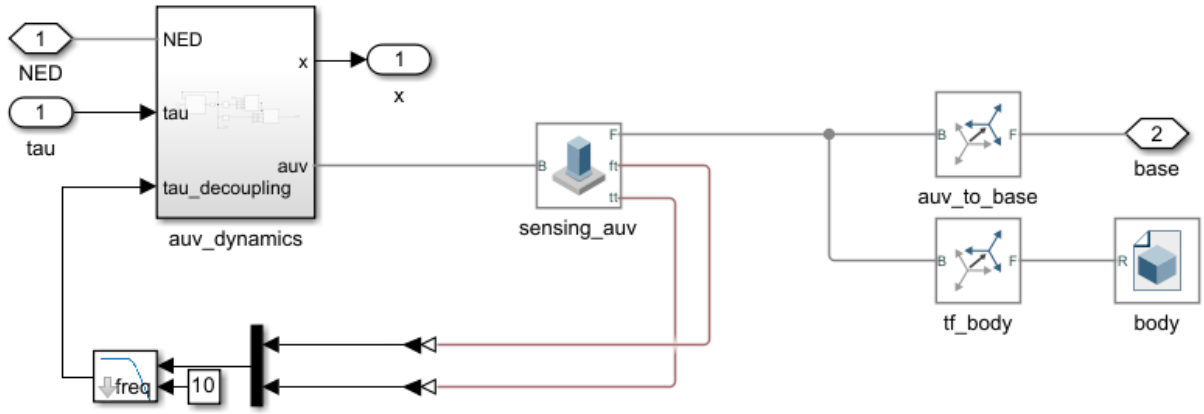


Figure 21: Simulink model of the AUV.

The lowpass filtering removes undesired high frequencies from the disturbance, with a cutoff frequency of 10 rad/s. Furthermore, *Rigid Transform* was used to place the visualization of the AUV from the *File Solid* block and create the base frame for the robot arm.

6.4.2 Robot Arm

Appendix C presents the structural overview of the robot arm model, showcasing where Simscape particularly excels. The frame transformations between various links and joints can be set up intuitively using *Rigid Transformation*, with the DH-parameters as a reference for the inputs. The different links of the Reach Alpha 5 are interconnected using

Revolute Joint blocks, which are actuated by torque and provide feedback on position and velocity. Spring stiffness and damping are provided as inputs to the *Revolute Joint* block, but friction is constructed separately using the *Rotational Friction* block.

Figure 22 shows the content of the subsystem block *link1* from the robot arm model. Each link contains the same elements. The *Inertia* block includes the description of mass and inertia, and the *File Solid* block includes the .STEP file with a visual description of the body and buoyancy is an *External Force and Torque* block which can be configured to consistently apply the force in the direction of the world z-axis.

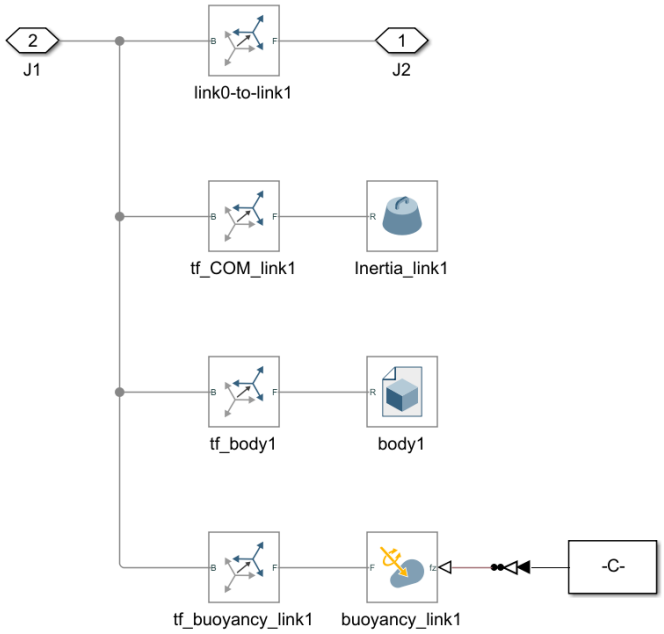


Figure 22: Contents of the subsystem block *link1* from Appendix C.

7 Control Implementation

This section will present the control methodology for the UVMS with the task of grasping dead fish. Figure 23 depicts the flow of the control architecture. A brief summary of the control methodology follows: A proposed kinematic control strategy is implemented to generate reference velocities, which are subsequently fed into a reference model. The reference model is of second order and generates smooth desired velocities and acceleration trajectories. The desired velocities are then compared with the system velocities to identify the velocity error. This error is multiplied by the proportional gain, and its integral is multiplied by the integral gain in the PID controller. The desired accelerations are fed directly to the derivative gain. The PID controllers compute desired forces and moments, which are fed to the system. Each step of the control design will be explained in more detail in the following sections.

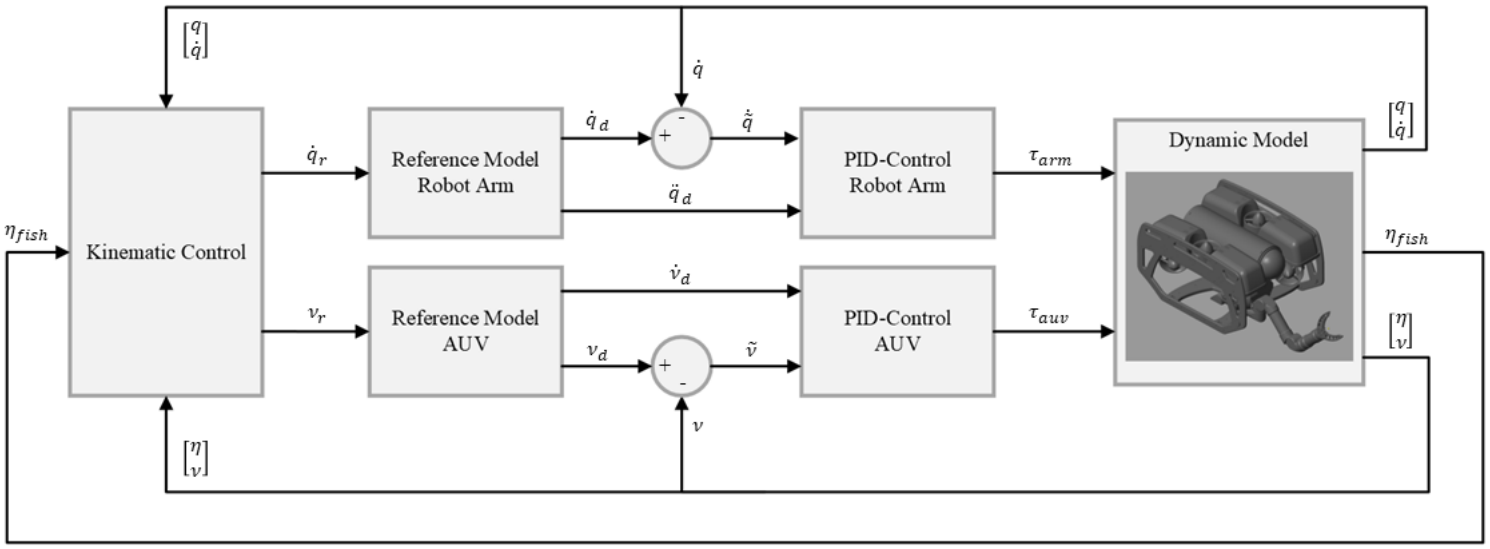


Figure 23: Control diagram of the proposed control approach.

7.1 Proposed Kinematic Control Law

The proposed kinematic control law is designed to enhance the operational efficiency of a UVMS tasked with grasping dead fish, a problem characterized by the need for precision and adaptability in a dynamic underwater environment. Inspired by the task priority framework in Section 5.1, this law innovatively employs a weighted transpose of the Jacobian matrix to manage the system's DOF effectively. The approach is divided into three primary tasks:

1. **Positioning AUV and keep fish in FOV:**
   The first task mandates the AUV to align itself to the y-axis of the fish to ensure a direct convergence to the long side of the fish while maintaining the fish within FOV to ensure continuous detection.

2. **Converge the end-effector to the grasping point:**
   Following the AUV's alignment, the goal is to converge the end-effector to the desig-

nated grasping point with collective DOFs of the UVMS without colliding or inducing disruptions to the floating fish.

3. **Manipulability optimization:**
   Preserving manipulability in the robot manipulator is crucial, as it ensures the maximum range of motion is available at all times to perform the assigned tasks effectively.

To adeptly manage these tasks, a dynamic prioritization strategy is implemented through a gain, Λ, effectively modulating the control effort between AUV positioning and end-effector convergence based on real-time assessments and operational needs. While task 3 is always providing control effort to maintain manipulability in the robot arm. The control vectors for the AUV and the robotic arm are defined as

$$\boldsymbol{\zeta}_r = \begin{bmatrix} \boldsymbol{\nu}_r \\ \dot{\boldsymbol{q}}_r \end{bmatrix} = \Lambda \mathbf{J}_1^T \mathbf{K}_1 \tilde{\boldsymbol{\sigma}}_1 + (1-\Lambda)\mathbf{J}_{2,W}^T \mathbf{K}_2 \tilde{\boldsymbol{\sigma}}_2 + \boldsymbol{\zeta}_{r,3} \tag{59}$$

Task 1 and 2 have their own error dynamics, $\tilde{\boldsymbol{\sigma}}_1$ and $\tilde{\boldsymbol{\sigma}}_2$, derived in Section 7.1.1 and 7.1.2. The error in Task 1 contains the position and yaw error of the AUV, while the error dynamics in Task 2 contains the pose error of the end-effector. Therefore, the Jacobians differ such that it corresponds to the different tasks. The Jacobian in task 1,

$$\mathbf{J}_1 = \begin{bmatrix} \boldsymbol{J}_{\text{auv}} & \mathbf{0}_{4\times 4} \end{bmatrix} \tag{60}$$

maps the system velocities to the motion of the AUV in x, y, z, and ψ, where the only contribution is the actuation from the AUV represented in $\boldsymbol{J}_{\text{auv}}$. Furthermore, $\mathbf{J}_2$ maps the system velocities to the end-effector velocities in all DOFs. Additionally, $\mathbf{J}_{2,W}^T$ is a weighted transpose with the weighting matrix $\mathbf{W}$ to weight the different system's DOF. The weighting vector $\mathbf{K}_1$ and $\mathbf{K}_2$, weights the different errors. These weighting matrices give the option to tune the system's response. $\mathbf{W}$ and $\mathbf{K}_2$ are dynamically adjusted to address additional subtasks and challenges, further elaborated in 7.1.3 and 7.1.4. Task 3 is derived in Section 7.1.5, using change in the manipulability index to determine velocity references that optimize manipulability.

The implementation of task functions in MATLAB is presented in the following sections. The implementation of the complete kinematic control is found in Appendix D.

7.1.1 Positioning AUV and keep fish in FOV

This section presents the error calculations and desired velocities for the AUV in task 1. The primary objective is to determine the desired velocities in x,y,z, and yaw (ψ) such that the drone is oriented perpendicular to the long side of the fish while maintaining the fish within the FOV to ensure continuous detection. The positioning is performed with linear velocities while keeping the fish within FOV with angular velocity in yaw.

To position the AUV such that it is aligned to the long side of the fish, it needs to converge to the y-axis of the fish. Therefore, the closest distance to this axis is found using the perpendicular distance

$$\boldsymbol{p}_e = \boldsymbol{v}_{\text{rel}} - (\boldsymbol{v}_{\text{rel}} \cdot \boldsymbol{y}_{\text{fish}})\boldsymbol{y}_{\text{fish}} \tag{61}$$

where $\boldsymbol{v}_{\text{rel}}$ signifies the relative position vector calculated as

$$\boldsymbol{v}_{\text{rel}} = \boldsymbol{p}_{\text{fish}} - \boldsymbol{p}_{auv}. \tag{62}$$

The perpendicular distance, $\boldsymbol{p}_e$, represents the distance and direction to the y-axis and can, therefore, be used as the positional error in the error dynamics of task 1. Furthermore, to keep the fish within the FOV, the AUV's camera should face the fish. This is achieved by calculating the desired yaw angle based on the relative position of the fish to the AUV as

$$\psi_{d,\text{auv}} = \text{atan2}(y_{\text{rel}}, x_{\text{rel}}) \tag{63}$$

where x_{rel} and y_{rel} is the x and y component of $\boldsymbol{v}_{\text{rel}}$. The error can then be calculated as

$$\psi_{e,\text{auv}} = \psi_{d,\text{auv}} - \psi_{\text{auv}} \tag{64}$$

resulting in the error dynamics for task 1, $\tilde{\boldsymbol{\sigma}}_1 = [\boldsymbol{p}_e, \psi_{e,\text{auv}}]^T$.

The transitioning between task 1 and task 2 is done with the gain, Λ. This gain is determined by the amplitude of the error in positioning and orientation by using the norm of the error in the sigmoid function [10],

$$\Lambda = \text{sigmoid}(\tilde{\boldsymbol{\sigma}}_1, e_0, \lambda) = \frac{1}{1 + \exp(12\frac{\|\tilde{\boldsymbol{\sigma}}_1\| - e_o}{\lambda - e_0} + 6)} \tag{65}$$

where $e_0 = 0.05$ and λ is a coefficient that can be used to tune the response of the sigmoid function explained in Section 5.1.6, which leads to a change in system response.

7.1.2 End-Effector Error Calculation

The end-effector position goal is obviously the grasping point, but the orientation is not straightforward. The desired roll can be found by examining the pitch of the fish but will depend on whether the system is converging on the left or right side of the fish. For simplification, the simulation test only converges on the left side of the fish, resulting in

$$\phi_d = -\theta_{\text{fish}} \tag{66}$$

The desired pitch angle is found by calculating the angle between the end-effector's position and that of the fish, yielding

$$\theta_d = \frac{-\text{asin}(z_{\text{fish}} - z_{\text{ee}})}{\|\boldsymbol{p}_{\text{fish}} - \boldsymbol{p}_{\text{ee}}\|} \tag{67}$$

where z_{fish} and z_{ee} are the z-coordinates of the fish and the end-effector, respectively, and $\boldsymbol{p}_{\text{fish}}$ and $\boldsymbol{p}_{\text{ee}}$ represent their position vectors. This methodology ensures the end-effector remains oriented towards the fish. Such an orientation is crucial for keeping the fish within the FOV of the camera mounted on the robot arm. Additionally, it minimizes the risk of the end-effector colliding with the fish.

Finally, the desired yaw of the end-effector is found as

$$\phi_d = \phi_{\text{fish}} + \frac{\pi}{2} \tag{68}$$

such that the angle of attack is 90 degrees on the fish's long side. This also depends on the side from which the system converges, but this solution does not address it.

Finally, this results in the desired end-effector pose, $\boldsymbol{\sigma}_d = \begin{bmatrix} \boldsymbol{p}_{\text{fish}} \\ [\phi_d, \theta_d, \psi_d]^T \end{bmatrix}$, such that the end-effector error can be calculated as

$$\tilde{\boldsymbol{\sigma}}_2 = \boldsymbol{\sigma}_d - \boldsymbol{\sigma}, \tag{69}$$

where $\boldsymbol{\sigma}$ is the end-effector pose.

7.1.3 End-effector Orientation Prioritization, K

The vector $\mathbf{K}_2 = [K_1, K_2, \ldots, K_6]^T$ is used to weight the control of each DOF of the end-effector. A higher value in the corresponding index of the vector will create harder control on this specific DOF. A suitable general weighting, $\boldsymbol{k}_1 = [k_{1,1}, k_{1,2}, \ldots, k_{1,6}]^T$, is selected to give reasonable response, but from testing it is found that more precise end-effector orientation is beneficial when the system converges towards the fish, and as the end-effector is closing in on the object, positioning becomes more important. Therefore, an adaptive change in the weighting was developed such that

$$\boldsymbol{k}_{2,[1:3]} = 1 - \Lambda_1 \tag{70a}$$
$$\boldsymbol{k}_{2,[4:6]} = 1 + \Lambda_1 \tag{70b}$$

where

$$\Lambda_1 = \text{sigmoid}(\tilde{\boldsymbol{\sigma}}_{2,[4:6]}, e_0, \lambda_1). \tag{71}$$

If the error in orientation is large, control in orientation is prioritized while control in position is deprioritized, using the gain Λ_1 from the sigmoid function. λ_1 is used to adjust the response of the function.

Finally, the weights in vector $\mathbf{K}_2$ can be calculated as

$$K_i = k_{1,i} \cdot k_{2,i} \tag{72}$$

7.1.4 Joint Limits & AUV vs. arm, W

The weighting matrix $\mathbf{W} = \text{diag}([W_1, W_2, \ldots, W_8])$ is used to weight the usage of each DOF within the system. For example, larger values in $W_{1:4}$ will reduce the usage of the AUV. A suitable general weighting, $\boldsymbol{w}_1 = [w_{1,1}, w_{1,2}, \ldots, w_{1,3}]^T$, is selected to provide a reasonable response before different subtasks alter the weighting depending on the current situation of the system.

Joint Limits

A solution to create weighting based on joint limits is found in [24] where

$$\frac{\partial \boldsymbol{H}(\boldsymbol{q})}{\partial q_i} = \frac{1}{\lambda_3} \frac{(q_{i,\max} - q_{i,\min})(2q_i - q_{i,\max} - q_{i,\min})}{(q_{i,\max} - q_i)^2 (q_i - q_{i,\min})^2} \tag{73}$$

provides an indication of each joint proximity to the limits. The coefficient λ_3 decides the sensitivity of the control system to joint limit proximity, effectively setting the strictness of adherence to these constraints. Then, the following logic

$$w_{2,i+4} = \begin{cases} 1 + \left\| \frac{\partial \boldsymbol{H}(\boldsymbol{q})}{\partial q_i} \right\|, & \text{if } ((\frac{\partial \boldsymbol{H}(\boldsymbol{q})}{\partial q_i} \text{ and } \dot{q}_i) > 0) \text{ or } ((\frac{\partial \boldsymbol{H}(\boldsymbol{q})}{\partial q_i} \text{ and } \dot{q}_i) < 0) \\ 1, & \text{otherwise} \end{cases} \tag{74}$$

is used to create the weighting on the different joints of the robot manipulator, such that it only penalizes movement towards the joint limit. Resulting in $\boldsymbol{w}_2 = [1, 1, 1, 1, w_{2,5}, w_{2,6}, w_{2,7}, w_{2,8}]^T$. Implemented as follows:

```
function [w] = joint_limits(q, q_dot, lambda)
    % INPUT
    % q = joint angles
    % q_dot = joint velocities
    % lambda = tuning coefficient

    q_max = [deg2rad(350);
             deg2rad(200);
             deg2rad(200);
             deg2rad(330)];

    w = ones(8,1);
    for i = 1:4
        H = ( 1/lambda ) * ( ( q_max(i) * (2*q(i)-q_max(i)) ) ...
            / ( (q_max(i)-q(i))^2 * (q(i))^2 ) );
        if ((H > 0) && (q_dot(i) > 0)) || ((H < 0) && (q_dot(i) <
           0))
            w(i+4) = 1 + abs(H);
        else
            w(i+4) = 1 + abs(.2*H);
        end
    end
end
```

Listing 1: Joint Limits Function

AUV vs. arm

Since the manipulator has less inertia and faster dynamics, it is beneficial to prioritize the usage of the manipulator when the end-effector is close to the object. Therefore, a weighting based on this idea is created such that

$$\boldsymbol{w}_{3,[1:4]} = 1 - 0.3\Lambda_2 \tag{75a}$$

$$\boldsymbol{w}_{3,[5:8]} = 1 + 0.3\Lambda_2 \tag{75b}$$

where

$$\Lambda_2 = \text{sigmoid}(\boldsymbol{\sigma}_{e,[1:3]}, e_0, \lambda_2) \tag{76}$$

such that the usage of the AUV will be reduced when the end-effector is close to the fish, and the usage of the arm will be increased. This also creates an adaptive solution such that the AUV can move faster when the system is further away from the fish, reducing the run time.

Finally, the weights of the system's DOF, W_i, are calculated as

$$W_i = w_{1,i} \cdot w_{2,i} \cdot w_{3,i} \tag{77}$$

7.1.5 Manipulability Optimization

To implement the task priority framework, one approach considered was the set-based manipulability tasks for kinematic singularity avoidance, as detailed in [28], see Section 5.1.5. Although this method was tested, it produced undesirable results, further discussed in Section 8. Consequently, a different strategy was proposed. This alternative method employs gradient-based optimization to assess how minor adjustments in each joint angle influence the manipulability measure. This approach generates velocity references designed to enhance the manipulability of the robot arm by using the manipulability index from Section 2.4.1. The manipulability optimization is implemented as follows:

```
function [zeta_r, w] = manipulability_optimization(q)
    % INPUT
    % q = joint angles

    J = compute_jacobian(q);
    Jp = J(1:3, :);

    W = Jp * Jp';
    w = sqrt(det(W));

    gradient_w = zeros(size(q));
    epsilon = 1e-6; % Perturbation for finite difference

    % Compute gradient using finite differences
    for i = 1:length(q)
        q_perturbed = q;
        q_perturbed(i) = q_perturbed(i) + epsilon;
        J_perturbed = compute_jacobian(q_perturbed);
        Jp_perturbed = J_perturbed(1:3, :);
        W_perturbed = Jp_perturbed * Jp_perturbed';
        w_perturbed = sqrt(det(W_perturbed));

        gradient_w(i) = (w_perturbed - w) / epsilon;
    end

    Lambda = sigmoid(w, 5.5e-3 , 6.5e-3);
    K = 20;
    q_dot_r = (1-Lambda)*K*gradient_w;
    zeta_r = [zeros(4,1); q_dot_r];
```

Listing 2: Manipulability optimization function.

The sigmoid function is used to define a region of acceptable manipulability measure. The gain K is used to tune the generated velocity references. This algorithm was made with the help of the AI model ChatGPT.

7.2 Reference Model

A second-order reference model is used to obtain smooth reference signals for the desired velocity [22]. This is done by using a second order LP-filter

$$\frac{x_d(s)}{x_{ref}(s)} = \frac{\omega_i^2}{s^2 + 2\zeta_i\omega_i s + \omega_i^2} \tag{78}$$

where ζ_i are the relative damping ratio and ω_i are the natural frequency for the i^{th} state.

The reference model is implemented as a first-order linear system

$$\dot{\boldsymbol{x}}_d = \mathbf{A}_d\boldsymbol{x}_d + \mathbf{B}_d\boldsymbol{r} \tag{79}$$

where

$$\mathbf{A}_d = \begin{bmatrix} \mathbf{0}_{n\times n} & \mathbf{I}_n \\ -\mathbf{\Omega}^2 & -2\mathbf{\Delta}\mathbf{\Omega} \end{bmatrix}, \quad \mathbf{B}_d = \begin{bmatrix} \mathbf{0}_{n\times n} \\ \mathbf{\Omega}^2 \end{bmatrix} \tag{80}$$

and

$$\mathbf{\Omega} = \text{diag}(\omega_1, \omega_2, \ldots, \omega_i) \tag{81a}$$
$$\mathbf{\Delta} = \text{diag}(\zeta_1, \zeta_2, \ldots, \zeta_i) \tag{81b}$$

With the reference signal, $\boldsymbol{r}$, as the reference velocities computed from the kinematic control. The resulting $\boldsymbol{x}_d$ will include a filtered velocity and acceleration reference.

The implementation of the reference model is as follows:

```
function [xd_dot] = sec_ord_ref_mod(xd, r, Omega, Delta)
    % INPUT
    % xd    = desired states
    % r     = reference states
    % Omega = diagonal damping matrix
    % Delta = diagonal natural freq matrix

    n = size(r,1);

    Ad = [zeros(n,n), eye(n);
          -(Omega^2), -2*Delta*Omega];
    Bd = [zeros(n,n);
            Omega^2];

    xd_dot = Ad*xd + Bd*r;
end
```

Listing 3: 2. order reference model Function

and used in Simulink as shown in Figure 24.

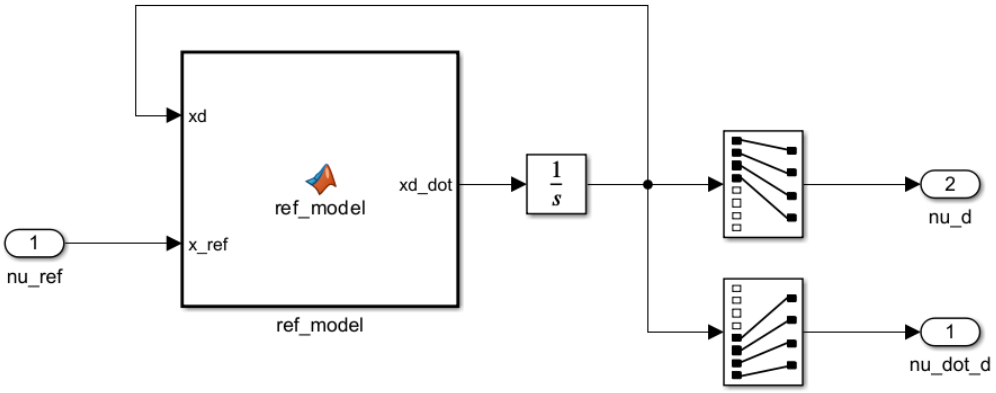


Figure 24: Refrence Model implementation for the AUV in Simulink.

7.3 PID Control

PID control is employed to control the velocity of each individual DOF of the system. The desired velocities from the reference model are compared with the system's velocities to find the velocity error used for the proportional and integral gain. Since robotic systems typically contain lots of noise in the acceleration feedback, the desired acceleration generated in the reference model is used directly with the derivative gain in the controller. The PID controller is implemented as a MATLAB function with an integral limit as follows:

```
function [u] = PID_controller(K, e, e_dot, int, int_limit)
% INPUT
% K         = [Kp,Ki,Kd]
% e         = error
% e_dot     = derivative error
% int       = integral of error
% int_limit = limit for the integral of error to prevent wind-up

Kp = K(1); Ki = K(2); Kd = K(3);

% Check and limit the integral part to prevent wind-up
if int > int_limit
    int = int_limit;
elseif int < -int_limit
    int = -int_limit;
end

% Calculate the control signal
u = Kp*e + Ki*int + Kd*e_dot;
end
```

Listing 4: PID Controller Function

It was implemented in Simulink as shown in Figure 25.

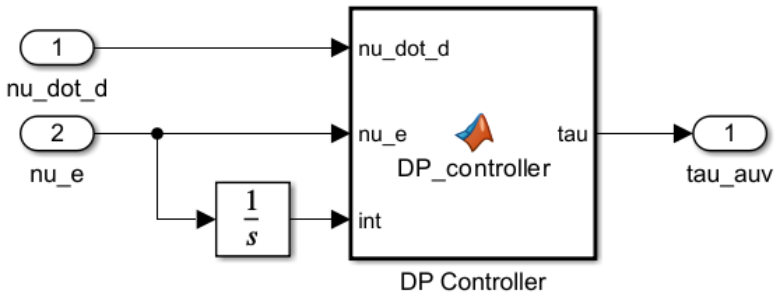


Figure 25: PID controller implementation for the AUV in Simulink.

8 Results

This section presents the simulation outcomes, specifically focusing on the tuning approach and the resultant data. The case that is examined initializes the AUV's state at $\boldsymbol{\eta} = [0, 0, 0, 0, 0, 0]^T$, and the fish's state at $\boldsymbol{\eta}_{\text{fish}} = [1.3, 0.5, 0, 0, 20°, 70°]^T$. This creates a scenario designed to challenge all system DOFs and necessitate a comprehensive coordination of the different tasks to grasp the fish successfully.

For this simulation, certain assumptions have been made to focus the scope of the study and ensure the clarity of the results:

- Fish pose is known.
- UVMS states are known.

8.1 Dynamic Control Tuning

The controller includes multiple parts that need tuning. The tuning approach first addresses the inner control loops, such as the PID control and reference models, before tuning the kinematic control.

8.1.1 Reference Model and PID controller Tuning

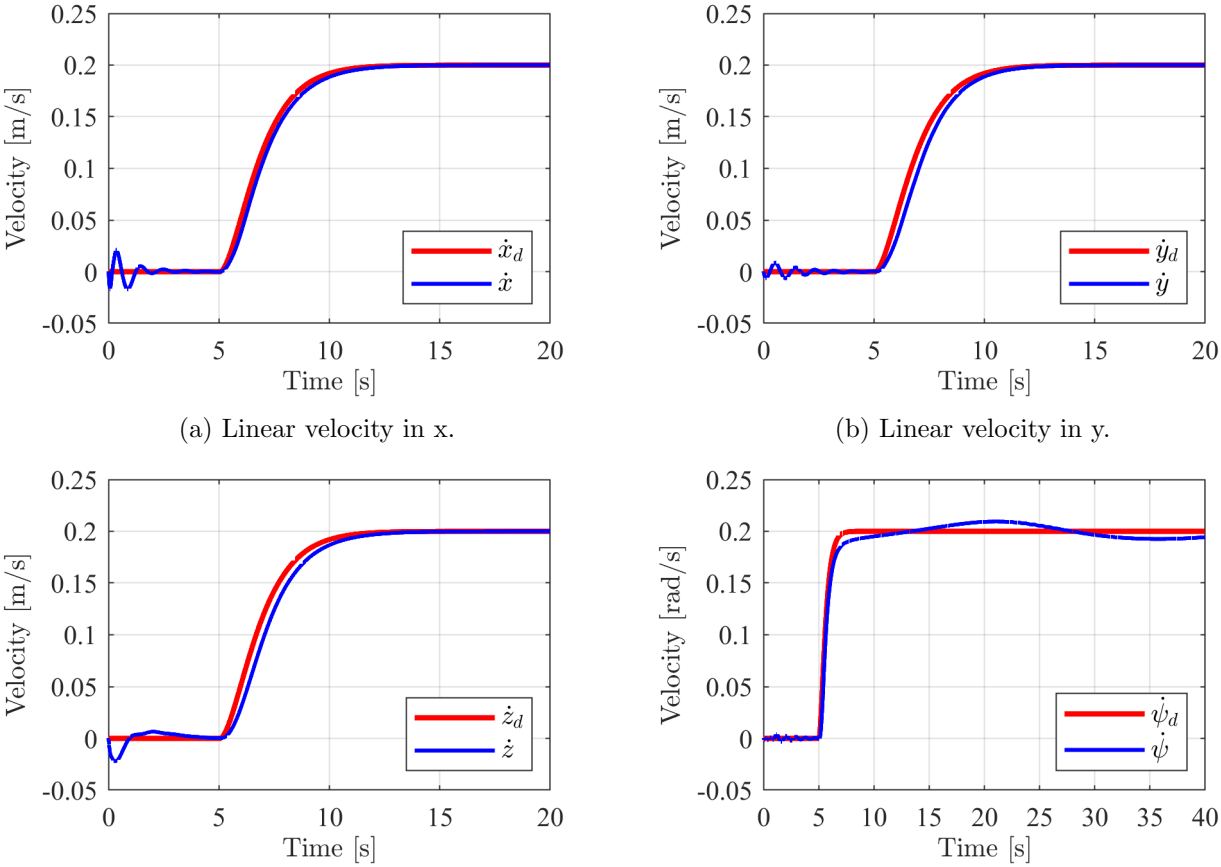
The reference model is present to generate a smooth trajectory for the desired velocities. When selecting the parameters for the reference model, it is crucial to consider both the desired system responsiveness and the practical response capabilities within the given DOF. It is important to balance these parameters to ensure the system does not move too quickly, which would increase hydrodynamic effects and influence on the floating fish, nor too slowly, which would extend the time it takes to catch the fish.

The underwater drone is capable of a maximum speed of 1.5 m/s. However, this is unreasonably fast for the task at hand. Therefore, a maximum velocity of 0.2 m/s is proposed. With no prior experience operating the drone, it is assumed that the BlueROV2 can achieve this velocity within 5 seconds without generating excessive hydrodynamic effects or disturbing the surrounding water. With a natural frequency and damping ratio equal to 1, the fully damped reference model generated suitable characteristics for the desired AUV velocities for the task.

The PID controller was tuned on a leap from 0 to 0.2, using the reference model to generate velocity trajectory. The parameters are listed in Table 3, and the results are shown in Figur 26.

Table 3: Reference model parameters and PID controller gains for the AUV.

Reference Model			**PID controller**				
DOF	$\boldsymbol{\omega}$	$\boldsymbol{\zeta}$	**DOF**	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	**integral limit**
x, y, z	1	1	x, y, z	40	30	10	10
yaw	3	1	yaw	2.5	2	0.2	1



(a) Linear velocity in x.

(b) Linear velocity in y.

(c) Linear velocity in z.

(d) Angular velocity in yaw.

Figure 26: Linear and angular velocity control response of the AUV in x, y, z, and yaw.

The linear states in Figures 26a, 26b, and 26c show promising results, demonstrating a stable and smooth motion of the AUV without overshooting, which could otherwise lead to undesired collisions with the object during convergence. Figure 26d exhibits some slow oscillations due to the decoupling in the AUV's dynamics. Since the AUV is not expected to rotate continuously at a constant speed, these results validate the control system's effectiveness.

For the robot arm mounted on the AUV, the reference model and PID control were explicitly tuned to leverage the arm's quicker response capabilities, taking advantage of its lower inertia compared to the AUV. This tuning strategy ensures that the reference model accurately reflects the robot arm's responsive behavior. Unlike the AUV, which exhibits slower dynamics and takes longer to reach operational speeds, the arm can react more swiftly, allowing for rapid adjustments.

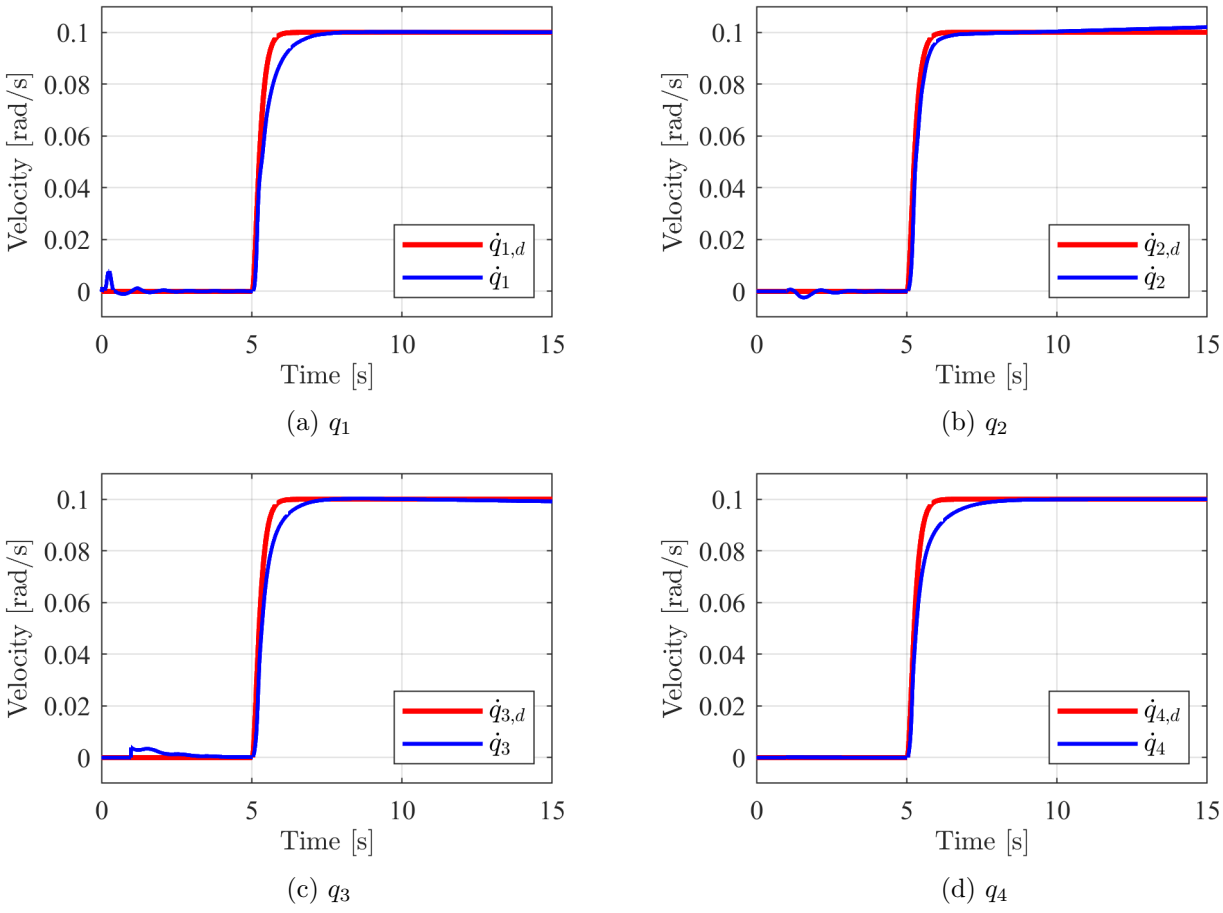
This responsiveness is beneficial for grasping dead fish, where precise and timely movements are necessary. By optimizing the arm's control to enable faster responses, the system can effectively leverage the arm's lower inertia to compensate for minor errors quickly. Since underwater conditions constantly introduce slight positional inaccuracies to the floating UVMS, this capability is essential for significantly increasing the operation's success.

The parameters used are listed in Table 4, and the response is shown in Figure 27.

Table 4: Reference model parameters and PID controller gains for the robot arm.

Reference Model		
DOF	ω	ζ
$q_{1:4}$	7	1

PID controller				
DOF	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	**integral limit**
q_1	2	5	0.1	10
q_2	5	20	0.1	10
q_3	5	10	0.1	10
q_4	6	9	0.2	10



(a) q_1 (b) q_2

(c) q_3 (d) q_4

Figure 27: Angular velocity control response for the four joints of the robot arm.

The figure showcases how the robot arm can respond quicker than the AUV without introducing overshoot. However, a small undesirable result is observed in Figure 27b, where the state slowly drifts from the desired velocity. This occurs because the joint initially lifts the arm against gravity and then moves past the point where gravity contributes to increasing the velocity. This highlights a potential limitation of PID control on the robot arm. However, during operation, the system is not designed to maintain a constant velocity.

8.2 Kinematic Control Tuning

The kinematic control includes three main tasks and additional subtasks with the possibility of tuning the response of each task. Table 5 lists the different tuning coefficients, including a brief description.

Table 5: Kinematic control coefficients.

Coefficient	Value	Description
λ	0.5	choose between orientating and converging AUV
λ_1	1	prioritizing orientation of end-effector
λ_2	2	joint limits
λ_3	0.5	prioritizing AUV control or robot arm during convergence

8.2.1 Task 1

The response in task 1 is tuned with the vector $\mathbf{K_1}$ where the first three values gain the positioning errors and the fourth gains the yaw error. By neglecting the other tasks and generating velocity references only from task 1,

$$\boldsymbol{\zeta}_r = \mathbf{J}_1^T \mathbf{K}_1 \tilde{\boldsymbol{\sigma}}_1, \tag{82}$$

the response could be tuned. Figure 28 shows how the system performed where the blue arrow represents the AUV and its orientation in the xy-plane. The figure shows how the system is able to align itself to the y-axis of the fish while maintaining it within the FOV. Thus preparing it for the next task, converging the end-effector to the gripping point.

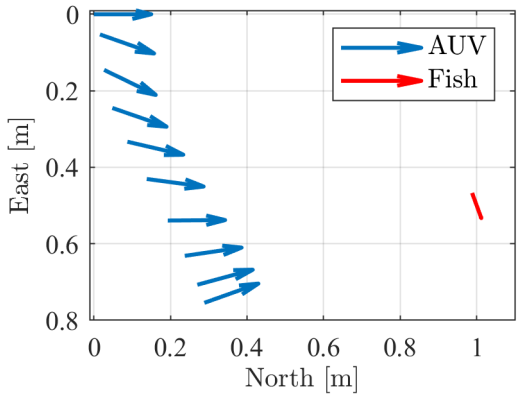
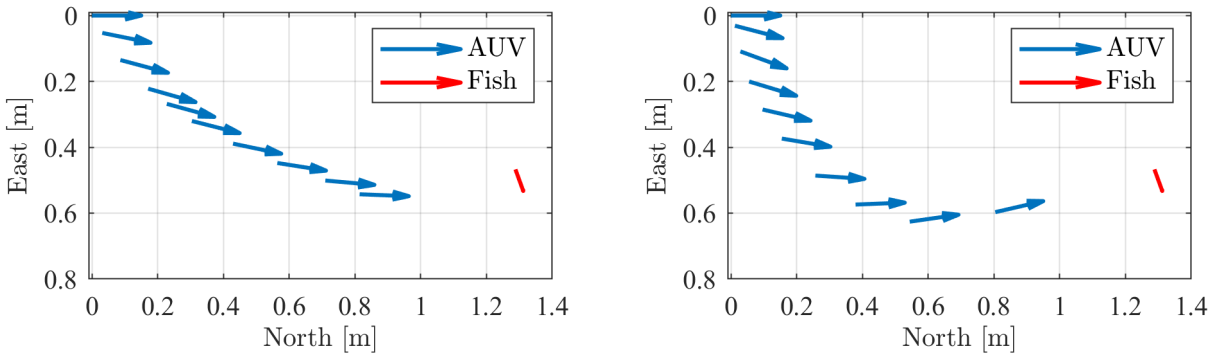


Figure 28: AUV positioning and orienting

8.2.2 Transition between Task 1 and Task 2

The system should be properly aligned to the long side of the fish before transitioning to task 2, which is to converge the end-effector to the grasping point. Figure 29 shows the motion of the AUV with two different values for λ. The coefficient is changing the sigmoid function, and in Figure 29a, with $\lambda = 2$, the system starts transitioning when the error norm is below 2. Compared with the results in Figure 29b where $\lambda = 0.5$, the system

creates a more optimized position and orientation before fully transitioning to task 2 and yielding a more robust and desirable result with the smaller λ.



(a) $\lambda = 2$ (b) $\lambda = 0.5$

Figure 29: AUV positioning and orienting with different transitioning coefficient λ.

8.2.3 Task 2

Task 2, converge the end-effector to the grasping point, includes multiple sub-tasks with individual tuning. This section will systematically explain the reasoning and tuning choices for each sub-task.

Orientation vs. position

In order to reduce the risk of undesired collision with the fish, it is crucial to have a suitable orientation relative to the fish before converging the end-effector to the grasping point. Figure 30 showcases when the robot arm does not have a suitable orientation before converging, increasing the risk of colliding the jaw with the fish.

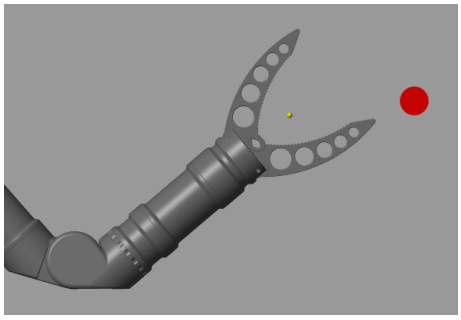
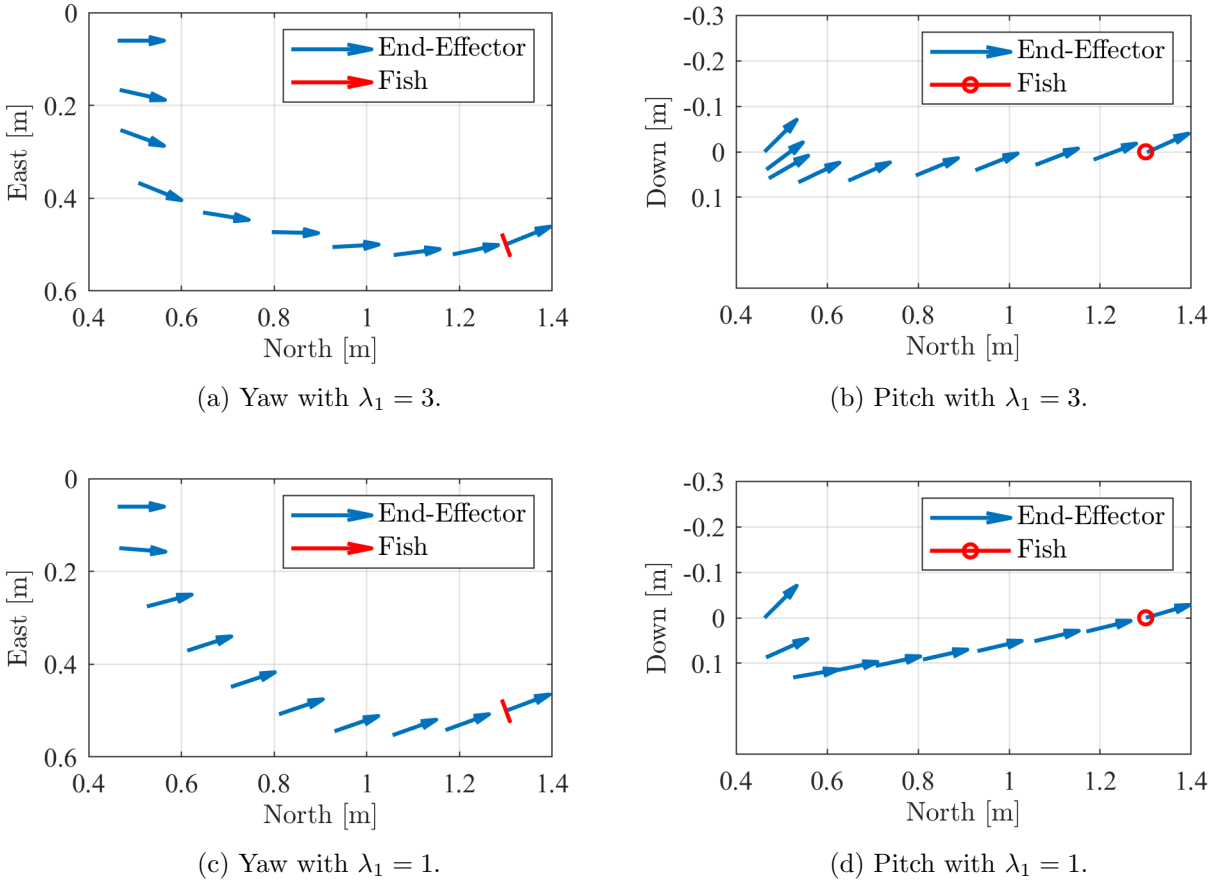


Figure 30: Example of a case where pitch is unsuitable before convergence. The red circle represents the fish.

Therefore, a prioritization between orientation and position was implemented. This was also conducted using the sigmoid function where λ_1 is used to tune the response. Figure 31 shows two cases with different λ_1, where the blue arrow represents the end-effector. Arguably, the response in Figure 31c and 31d yields the most suitable results, reducing the risk of collision and increasing the chance of a successful grasp. However, Figure 31a shows a suitable result for keeping the fish within FOV of the camera mounted on

the robot arm. Nevertheless, Figure 31b shows that the pitch becomes undesirable for convergence, thus outweighing the choice of a smaller λ_1. This result does, however, suggest an additional task of maintaining the fish within the FOV of the camera mounted on the robot arm could be implemented.



(a) Yaw with $\lambda_1 = 3$.

(b) Pitch with $\lambda_1 = 3$.

(c) Yaw with $\lambda_1 = 1$.

(d) Pitch with $\lambda_1 = 1$.

Figure 31: End-effector yaw and pitch with different λ_1.

Joint Limits

Including a task that restricts actuation near joint limits serves primarily as a safety feature rather than a core aspect of the control design. Nevertheless, it effectively prevented the arm from reaching a singular position during operation, with the absence of singularity avoidance or manipulability optimization. While integrating the additional tasks, the influence of the joint limits task became less noticeable.

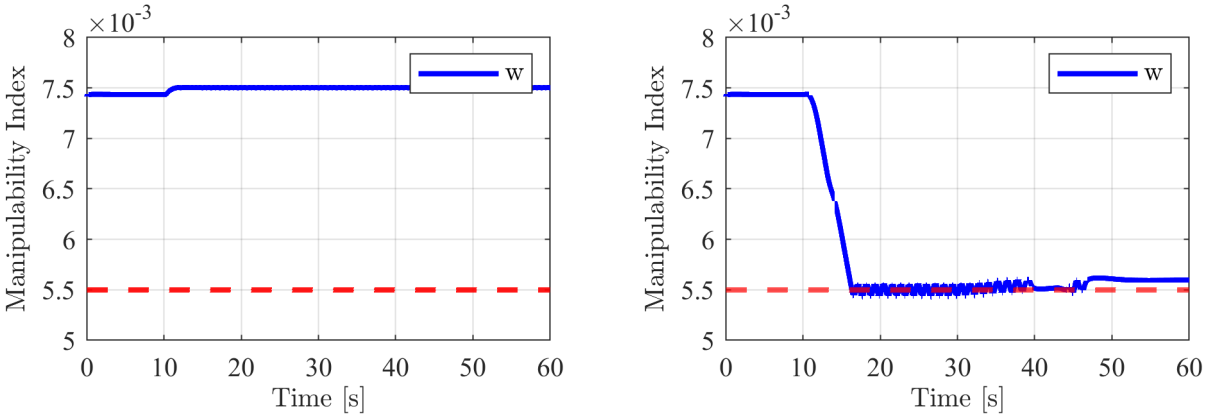
Tuning the joint limits task involved deciding the level of restriction near the joint limits, which directly affected the arm's responsiveness. Stricter tuning could lead to reduced responsiveness, presenting a trade-off between safety and performance. This consideration is essential for optimizing system behavior, ensuring that the arm remains both safe and effective.

8.2.4 Task 3

In the simulation scenario, the system must align itself before proceeding with the task. While the system is aligning, the robotic arm simultaneously attempts to converge on the fish. Due to the time it takes for the system to align properly, the arm starts to extend fully while still in the alignment process. This premature extension can lead the arm toward a fully stretched position, significantly reducing its manipulability. The joint limit task employed handles this issue to some extent. Still, as the manipulability index continued to decline, it became apparent that a more effective solution was required to prevent the arm from reaching these near-singular configurations and maintain its operational effectiveness throughout the task.

The singularity avoidance task, introduced earlier in Section 5.1.5, was tested in the simulation, resulting in undesirable results. Figure 32 compares the manipulability index with and without the set-based approach. The absence of a set-based approach, depicted in Figure 32a, reveals that the robotic arm trends toward a manipulability maximum. Projection to the null space continually deprioritizes the task of converging toward the gripping point, leading to unsuccessful convergence.

Figure 32b illustrates results using the set-based approach. An if-statement ensures that the singularity avoidance task remains inactive as long as the manipulability index stays above 5.5×10^3. This approach also led to suboptimal results, as the system frequently toggles the singularity avoidance task on and off, inducing oscillations within the system. Despite these challenges, the system ultimately converges to the gripping point.

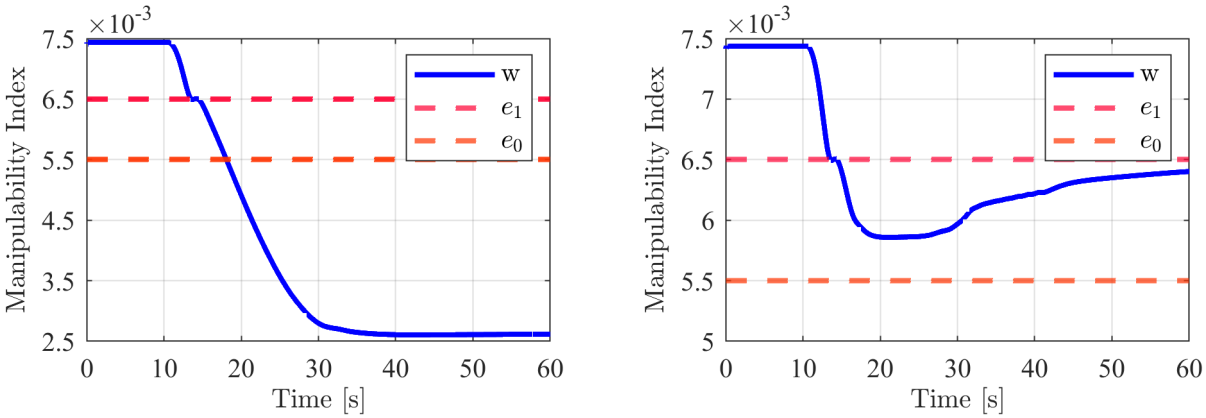


(a) Without set-based.

(b) With set-based.

Figure 32: Singularity avoidance with and without set-based approach.

The proposed gradient-based optimization approach yielded more favorable results. Figure 33 presents two plots: Figure 33a displays outcomes without the presence of manipulability optimization or singularity avoidance, illustrating the manipulability index converging to zero, indicating that the robot arm is approaching a singular configuration. However, it did not reach zero due to the intervention of the joint limit task. Conversely, Figure 33b demonstrates the effectiveness of the proposed manipulability optimization, maintaining a suitable manipulability index while minimally impacting the convergence process.



(a) Without manipulability optimization.

(b) With manipulability optimization.

Figure 33: Manipulability index with and without manipulability optimization.

8.3 System Performance

Figure 34 presents the task error $\tilde{\sigma}_2$ for Task 2, which involves the end-effector converging to the grasping point. The simulation allows a stabilization period of 10 seconds before initiating control efforts. This figure illustrates the end-effector's successful convergence to the designated grasping point. It also highlights the prioritization of orientation control within the system's approach. This is evident as the attitude errors approach zero earlier in the sequence, indicating that orientation adjustments are prioritized over positional corrections. The error in the x dimension is the last to be corrected, underscoring the system's strategy to stabilize the end-effector's orientation before fine-tuning its position.

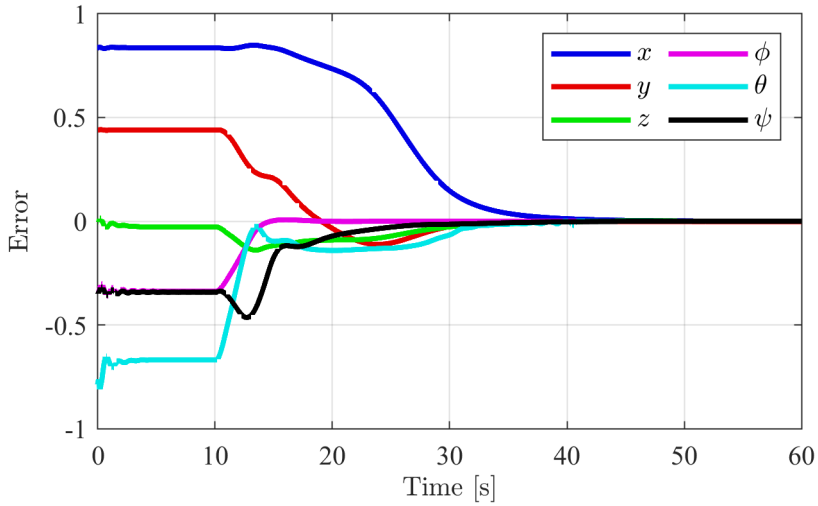


Figure 34: End-effector pose error.

9 Discussion

This section discusses the comprehensive results and insights gained from the research conducted on grasping dead fish underwater using a robot manipulator mounted on an underwater drone. This study primarily focused on control solutions, supplemented by an exploration of grasping theories and robot vision. Although grasping and robot vision were not the primary focus of this research, they provide an overview of the task and lay the groundwork for future investigations. A significant portion of the research involved creating and refining a simulation in MATLAB, which provided valuable lessons in handling CAD files and simulation settings. Additionally, the thesis explores various control strategies for UVMS and redundant systems, employing a novel approach inspired by, but not fully aligned with traditional task priority frameworks.

This section will first explore the benefits of the simulation created during the research and discuss how it informed various design choices. While simulations cannot replicate real-life conditions perfectly, they are crucial for validating task execution. The discussion will then link the pre-project phase and the development of the state machine to its impact on the research trajectory. Following this, the section will address the challenges encountered in implementing the control strategies, informed by the theoretical framework provided. Additionally, the outcomes of the simulation and the contributions of this research will be examined. Finally, it will discuss a theoretical analysis of grasping a slippery and non-rigid object, such as a dead fish, and the perceptual aspects of the task.

9.1 Simulation

The development of the simulator has been an educational experience, necessitating the exploration of new fields within CAD software and Simscape to build the model. The Simscape toolbox proved to be a good choice due to its extensive capabilities. It was highly intuitive and is recommended for others simulating multi-linked manipulators. Although time-consuming, establishing a robust platform for development has been crucial. Primarily, visual simulation motivates the development process. It breathes life into the research and adds an element of excitement. However, the benefits extend beyond entertainment; simulation offers intuitive insights into the system's behavior and movements, simplifying the troubleshooting process.

Several specific instances where the simulation has been instrumental in the development process can be highlighted. Firstly, it has enhanced the understanding of the workspace and maneuverability of the robot arm. During the construction of the robot arm and the positioning of joints and actuators, it became apparent that an elbow-down configuration provides better maneuverability than an elbow-up. Additionally, the simulation has identified the most practical placement for the arm on the AUV, considering space limitations due to the AUV and robot arm design. Moreover, the visual aid has been instrumental in detecting unrealistic motions during controller design, highlighting instances where the system is over-actuated or performing motions that may not be feasible in real-world scenarios.

One objective was to create a simulator that was as realistic as the task required, which was largely achieved. However, several areas could be improved with more time. Firstly, the drone lacks thruster dynamics. The thrust force reference is currently processed through a low-pass filter to simulate thrust inertia. This treatment is identical across all axes, which is unrealistic considering the drone's thruster configuration, which would produce

different possible forces in different axes. Additionally, the model does not account for hydrodynamic forces from added mass in the robot manipulator. As the task is performed at very low velocities, hydrodynamic effects can be neglected, but their inclusion could have enhanced the realism of the simulation. The reason for its absence is that a method to implement added mass within the Simscape toolbox was not identified. Lastly, significant noise factors have not been introduced. Thus, the simulation serves more as a validation of the methodology. In real-world applications, there will always be some uncertainty in the estimations regarding position and orientation and external noise affecting the system, introducing new challenges to the control design.

9.2 The System as a State Machine

In the pre-project [1], a state machine was developed to systematically structure the task of grasping dead fish, as illustrated in Figure 35. This model facilitated a structured approach to identify and address challenges at each phase of the task, providing a crucial foundation for the development process. Table 6 summarizes each state's primary functions and considerations, offering a concise overview of the system's operational logic from the pre-project.

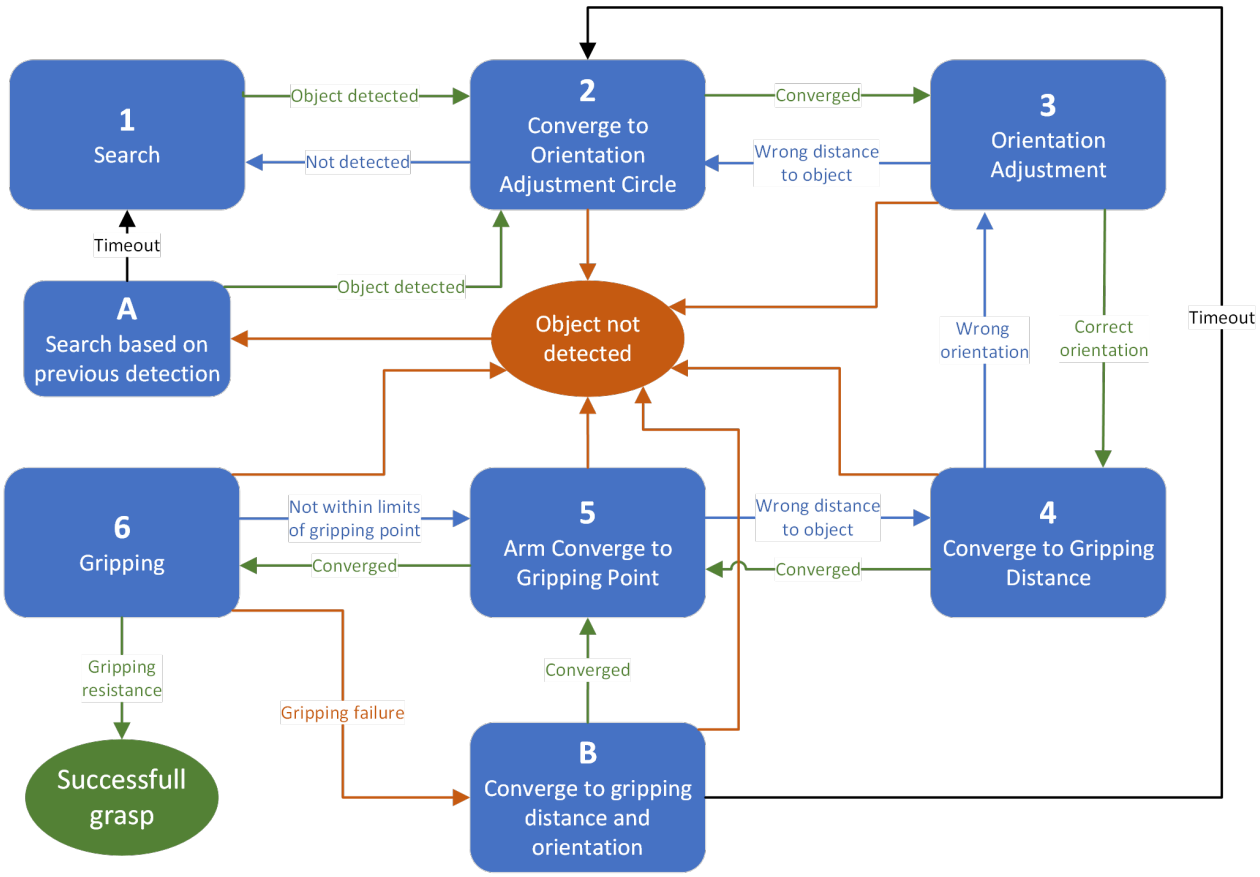


Figure 35: Proposed state machine for the problem.

Table 6: A brief summary of the different states in the state machine.

Index	**State**	**Description**
1	Search	Search pattern and detection of dead fish
2	Converge to Orientation Adjustment Circle	A circle with a suitable distance from the object to reduce influence
3	Orientation Adjustment	Following the circle until the robot arm is orientated to the longside of the object
4	Converge to Gripping Distance	Converge to a position that ensures maximum maneuverability for the robot arm
5	Arm Converge to Gripping Point	Robot arm aligning the end-effector with gripping point
6	Gripping	Actuation of the end-effector
A	Search based on previous detection	A more efficient search pattern when object detection is lost
B	Converge to gripping distance and orientation	A efficient relapse to position the vehicle correctly after a gripping failure

Subsequent analysis of each state revealed specific operational challenges and informed the design of a more nuanced control strategy. The original state machine, while effective for basic task structuring, evolved into a more fluid control model inspired by the task priority framework. This adaptation allowed asmoother transition between task phases and better alignment with real-time operational demands.

Key features developed from this reasoning include ensuring proper system alignment with the fish to enhance the success of converging and grasping maneuvers, originating from state **2** and **3**. It was identified that approaching the fish at an angle significantly complicates the task. Proper orientation adjustment in the AUV and the robot arm also contributed to reduced risk of undesired collision with the fish during convergence. Furthermore, the robot arm's maneuverability was initially maximized by positioning the drone with state **4**. This relies on a separate control approach using the AUV as a stationary base for the robot arm. Separated control depends on a large inertia ratio as mentioned in section 5.2. Therefore, utilizing a seamless kinematic control design using all available system DOFs of the redundant UVMS was more beneficial while continuously maintaining manipulability in the robot arm. Additionally, the control strategy accounts for the inertia differences between the arm and the underwater drone. By designing the control system such that the robot arm compensates for minor positional adjustments, it utilizes the lower inertia of the robot arm for faster error compensation.

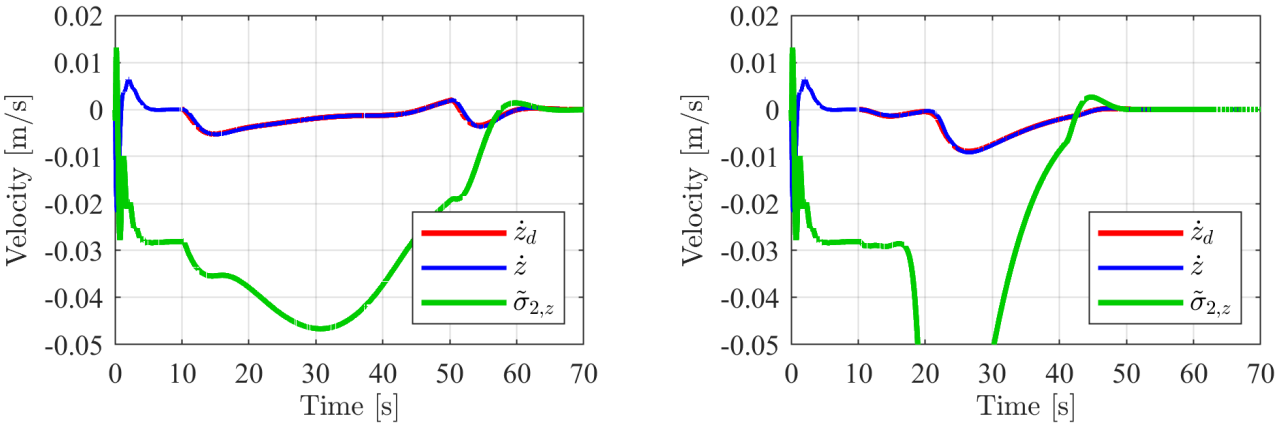
Although this thesis does not propose a complete solution for grasping dead fish, such as the state machine, it is essential to address the task comprehensively to gain a thorough understanding. As engineers, it is imprudent to focus solely on one specific aspect of the task. Integrating all components successfully requires an overview, which the development process of the state machine has provided. It has led to an evolved approach, where the control system not only adheres to the structural benefits of a state machine but also incorporates the flexibility and responsiveness necessary for complex underwater tasks.

9.3 Challenges with Control Implementations

One of the primary challenges encountered in this project involved the use of the pseudo-inverse of the manipulator Jacobian to solve inverse kinematics. Although generally effective, the approach faced issues during simulations. Specifically, an unexpected behavior was observed where the velocity reference for the AUV's z-axis moved in the opposite direction of the actual error in the end-effector's z-position. At time 50s, the task error $\tilde{\boldsymbol{\sigma}}_2(3) = -0.0195$, while the desired z-axis velocity $\boldsymbol{\zeta}_{r,2}(3)$ was calculated to be 0.0195, indicating a contradiction in motion direction as shown in the following calculation,

$$J_2^\dagger \tilde{\sigma}_2 = \begin{bmatrix} 0.9286 & -0.2003 & 0.0985 & -0.0003 & 0.0194 & -0.0392 \\ 0.2607 & 0.9435 & -0.0210 & 0.0003 & -0.0020 & -0.3396 \\ 0.0202 & 0.0033 & 0.8280 & -0.0000 & 0.2061 & -0.0000 \\ -0.1221 & 0.1725 & -0.0147 & -0.0017 & -0.0017 & -0.0002 \\ -0.1220 & 0.1723 & -0.0147 & -0.0017 & -0.0017 & -1.0002 \\ 0.4147 & -0.1599 & -1.1944 & -0.0001 & 0.1910 & -0.0000 \\ 0.4114 & -0.1552 & -1.1948 & -0.0002 & -0.8091 & -0.0000 \\ 0.0048 & -0.0068 & 0.0006 & -0.9999 & 0.0001 & 0.0000 \end{bmatrix} \begin{bmatrix} 0.0378 \\ -0.0279 \\ -0.0195 \\ -0.0128 \\ 0.1696 \\ 0.0104 \end{bmatrix} = \begin{bmatrix} 0.0417 \\ -0.0200 \\ 0.0195 \\ -0.0094 \\ -0.0198 \\ 0.0758 \\ -0.0940 \\ 0.0132 \end{bmatrix}.$$

Figure 36 showcases this issue, where Figure 36a shows how the AUV velocity in the z-axis is positive while the error is negative. Figure 36b shows the simulation results using the transpose of the Jacobian, yielding more suitable results. While the pseudo-inverse of the Jacobian is typically employed for its robustness against singularities, this study found that the transpose of the Jacobian was sufficiently adequate for the required tasks.



(a) Pseudo-inverse of Jacobian.

(b) Transpose of Jacobian.

Figure 36: Showcasing issues with pseudo-inverse of Jacobian.

The next challenge encountered involved the task-priority framework, specifically the component that projects onto the nullspace of the task Jacobian to allocate available DOF to subsequent tasks. This approach proved unsuitable for the designed tasks due to the Jacobian's persistent full-rank condition. Consequently, a sigmoid function was introduced to address this issue. This function allowed for effective transitioning between tasks by modulating the influence of different degrees of freedom based on the error norm, which proved beneficial for tasks such as Task 1 and Task 2 in this project, functioning akin to a seamless state machine.

Further attempts were made to utilize the task-priority framework when exploring the preservation of maneuverability in the robot arm. Specifically through a set-based method that initially seemed promising for this purpose, as similar methods have been employed

in previous studies focusing on singularity avoidance and manipulability preservation. However, the set-based task-priority approach operates by toggling the task on and off depending on whether the value falls within the predefined acceptable set, which led to oscillations due to frequent switching of the task status. This toggling also occurs in [28], but it does not seem to affect the system in the same manner. It indicates that the implementation used in this project may be incorrect. Although there exist methods that can dampen these oscillations, different approaches to manipulability preservation were explored. Consequently, a gradient-based optimization method was proposed, yielding suitable results. This method, developed with the assistance of OpenAI's ChatGPT, represents a novel approach not found in previous research literature. As engineers, it is advantageous to leverage all available tools to benefit the project. Incorporating an AI-driven algorithm like ChatGPT is part of the innovative approach, demonstrating how cutting-edge technology can be integrated into engineering solutions to enhance their effectiveness and adaptability.

9.4 Simulation Results

The simulation demonstrated the system's capability to effectively converge to the designated grasping point. It managed to do so without causing the jaws to collide with the object or overshoot the position, which would also result in undesired collisions. The system stabilized at the grasping point, utilizing all available degrees of freedom efficiently. The AUV and robotic arm collaborated effectively to achieve the objective. Although the simulation did not incorporate any form of noise or external disturbances, the results were positive, showing that the method could sensibly orient the system before proceeding with the convergence to the grasping point. This indicates that while the current control strategy is effective under ideal conditions, further testing under more realistic conditions incorporating disturbances is necessary to fully validate the robustness and practicality of the control system in real-world applications.

9.5 Dynamic Control

The control system implemented in the simulation utilized PID control for each DOF in the UVMS, achieving adequate results within the simulated environment. However, the occurrence of external noise and other factors in real-world applications may necessitate more robust and sophisticated control designs. Section 5.3 discusses several dynamic control methods that warrant further analysis and testing to determine the most appropriate approach for the specific task. Based on the research, sliding mode control has been frequently used in controlling UVMS due to its ability to counteract changes in parameters or environmental conditions adaptively.

9.6 Solution for Grasping Dead Fish

This section will discuss the challenges associated with detecting and estimating the pose of dead fish and the approaches to grasping them. This discussion is primarily theoretical, as no direct simulations or experimental tests have been conducted.

Underwater environments are notorious for their poor lighting conditions and limited visibility, presenting challenges for camera-based vision systems. These adverse conditions

often reduce the amount of detail captured in images, impacting the effectiveness of detection and pose estimation methods for robotic applications. Despite these challenges, the techniques discussed in Section 3 demonstrate promising potential for enhancing object recognition and spatial orientation in such environments. As the section outlines, advanced algorithms like YOLOv3 for object detection have already shown significant promise in similar underwater tasks. This method, which efficiently handles complex image analysis even under constrained conditions, could enhance the operational capabilities of a UVMS. However, the actual performance of these techniques must be tested under specific conditions when grasping dead fish to validate their practical applicability and reliability. This includes ensuring that the algorithms can handle low-light conditions and are robust against the variances in water clarity and color distortion commonly experienced underwater. Furthermore, the current system's reliance on monocular cameras may overcomplicate the task when equipping stereovision could be a cost-effective alternative. Nonetheless, developing methods to simulate depth perception using the two existing monocular cameras is feasible and warrants further testing for validation.

Grasping dead fish presents a significant challenge due to their non-rigid bodies and slippery surfaces. Additionally, the need to minimize damage during grasping further complicates the task. Using a form closure approach could minimize the impact on the fish, as it does not require force to maintain a stable grasp. However, the effectiveness of this approach is limited by the capabilities of the end-effector. The single-interlock jaw gripper currently used creates too few contact points for the complex geometrical shape of a fish. Nonetheless, as illustrated in Figure 37, point G2 at the fish's tail fin provides a geometric pit that might allow for effective form closure with the current jaw gripper. Alternatively, a force closure approach could be implemented, which depends on the angle of the contact point satisfying the friction cone to ensure a stable grip. Point G1 in Figure 37 might offer a suitable grasping point for force closure, as the jaw would contact a flat surface of the fish. This method would require a force feedback system to monitor the applied force, ensuring sufficient friction to reduce the likelihood of slippage while avoiding excessive pressure that could damage the fish. Although the current gripper lacks force sensors, it includes a sensor that measures its opening. Therefore, the applied force can be estimated by analyzing the electrical current supplied to the actuator and observing changes in the gripper's opening.

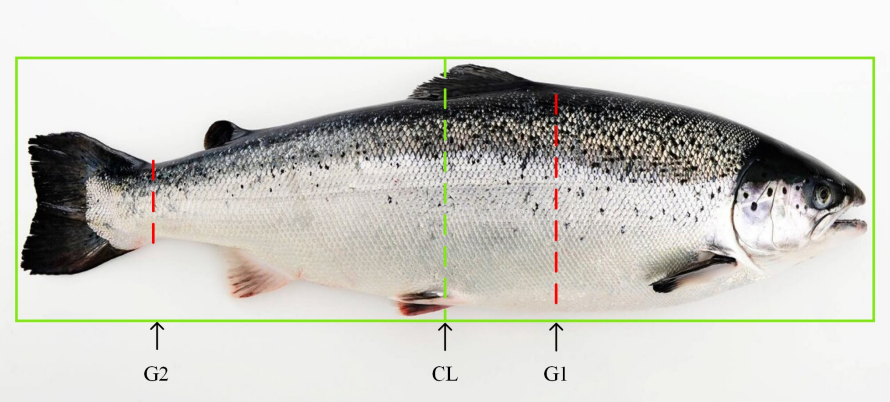


Figure 37: Example image of salmon and its optimal grasping points [34].

10 Conclusion

The simulation results demonstrate effective convergence, where the system is avoiding collisions and utilizing all available DOF to reach a stable grasping point. The simulation validates the control methodology, but further testing is needed for validation for real-world applications. The research also identifies YOLOv3 as a promising method for real-time object detection, and by using bounding boxes of the fish body and its head, the orientation can be estimated. Additionally, stereo-vision techniques can be applied to estimate position. Although promising, the methods need to be validated and tested before any conclusion of their functionality in the task of grasping dead fish can be made. In exploring grasping mechanisms, the study evaluated both form closure and force closure approaches. Form closure was found limited by the current end-effector design, which does not adequately conform to the complex shapes of fish due to its few contact points. Conversely, force closure showed potential, especially when enhanced by feedback systems that monitor and adjust gripping force to prevent slippage without damaging the fish. Grasping dead fish with a single-interlock jaw gripper may overly complicate the task. Therefore, changing to a different end-effector could significantly increase the success rate.

Pursuing an approach that leverages a low-cost platform is highly beneficial for the industry. By utilizing a versatile UVMS capable of performing various tasks at minimal expense, the technology becomes more feasible for investment and can more readily contribute to environmental improvements within the sector. The methods presented in this thesis propose such an approach with promising results. Nonetheless, real-world testing is essential to conclusively determine the system's functionality in the task of grasping dead fish.

Recommendation for Future Work

Robot Vision

The theory presented on robot vision introduces compelling methods for a low-cost UVMS that warrant further investigation. The use of monocular cameras on both the AUV and the robot arm to simulate stereo vision, coupled with the application of YOLOv3 for object detection and pose estimation targeting the fish's body and head, presents an intriguing approach. However, this strategy requires thorough investigation to validate its feasibility and effectiveness. Ideally, a pre-trained model specifically for farmed salmon should be utilized; if such a model is unavailable, developing this capability will also be necessary. Further research should focus on refining these technologies to enhance their accuracy and reliability in real-world underwater conditions.

Different end-effector on Reach Alpha 5

The simple interface of the Reach Alpha 5's end-effector presents opportunities to adapt or change it to a tool more suitable for the specific task of grasping dead fish. Switching to an end-effector that reduces the risk of slippage could significantly enhance the occurrence of a successful grasp and also reduce the risk of damaging the fish. For example, the soft grippers presented offer an intriguing starting point for further development and exploration.

Dynamic Control

Future research should explore alternative dynamic control strategies for the UVMS, especially in tasks like grasping dead fish, where this study primarily employed PID control. Investigating diverse control methods could significantly improve precision, stability, and efficiency, effectively tackling the unique challenges of underwater environments.

References

[1] Lasse Moen Guttormsen. 'Underwater Autonomous Grasping of Dead Fish'. Course: TTK4551 - Engineering Cybernetics. Specialization Project. Norwegian University of Science and Technology, Dec. 2023.

[2] Monica Dick. *Robotic Solutions for Aquaculture*. Aquaculture North America. 6th Jan. 2021. URL: https://www.aquaculturenorthamerica.com/robotic-solutions-for-aquaculture/ (visited on 16/09/2023).

[3] Christopher Noble, Kristine Gismervik, Martin Haugmo Iversen, Jelena Kolarevic, Jonatan Nilsson, Lars Helge Stien and James F. Turnbull. *Welfare Indicators for farmed Atlantic salmon : tools for assessing fish welfare*. Nofima, 2018.

[4] Bjørn Olav Nordahl. *Hver tredje Mowi-laks døde i merdene*. NRK. 30th June 2023. URL: https://www.nrk.no/dokumentar/hoy-laksedodelighet-hos-verdens-storste-oppdretter-1.16463576 (visited on 17/11/2023).

[5] Ole Reinert Omvik. *Mattilsynet mener syk og selvdød laks var på vei til forbrukerne*. NRK. 11th Oct. 2023. URL: https://www.nrk.no/norge/mattilsynet-mener-syk-og-selvdod-laks-skulle-selges-som-fersk-matfisk-1.16588744 (visited on 17/11/2023).

[6] Line Tomter. *Forsker om lekkede bilder av massedød: – Oi, dette var mye fisk*. NRK. 1st Nov. 2023. URL: https://www.nrk.no/norge/plutselig-dode-titusenvis-av-laks---laksegigant-ventet-en-uke-med-a-varsle-1.16601145 (visited on 17/11/2023).

[7] Michael Cantillon and Mads Ulrich Mjanger. 'Preliminary Design and Modeling of a System for Collecting Dead Fish in Offshore Aquaculture'. Bachelor thesis. Norwegian University of Science and Technology, May 2023.

[8] *Reach Alpha: Smaller Platform ROV Manipulator*. Reach Robotics. URL: https://reachrobotics.com/products/manipulators/reach-alpha/ (visited on 16/09/2023).

[9] *BlueROV2*. Blue Robotics. URL: https://bluerobotics.com/store/rov/bluerov2/ (visited on 16/09/2023).

[10] Mohammed Marey and François Chaumette. 'A new large projection operator for the redundancy framework'. In: International Conference on Robotics and Automation. IEEE, May 2010, pp. 3727–3732.

[11] Leonid B. Freidovich. *Modelling in Robotics and Control Methods for Robotic Applications*. Umeå University, Feb. 2021.

[12] Henrik Baldishol. 'Practical application of potential fields path planning on robot manipulator'. Master thesis. Norwegian University of Science and Technology, June 2022.

[13] Nikolaus Vahrenkamp and Tamim Asfour. 'Representing the robot's workspace through constrained manipulability analysis'. In: *Autonomous Robots* 38.1 (Jan. 2015), pp. 17–30.

[14] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*. 2nd ed. Cham: Springer International Publishing, 2016.

[15] Baohua Zhang, Yuanxin Xie, Jun Zhou, Kai Wang and Zhen Zhang. 'State-of-the-art robotic grippers, grasping and control strategies, as well as their applications in agricultural robots: A review'. In: *Computers and Electronics in Agriculture* 177 (Oct. 2020), p. 105694.

[16] Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi. 'You Only Look Once: Unified, Real-Time Object Detection'. In: *Cornell University Library, arXiv.org* (May 2016).

[17] Joseph Redmon and Ali Farhadi. 'YOLOv3: An Incremental Improvement'. In: *Cornell University Library, arXiv.org* (Apr. 2018).

[18] Bent Oddvar Arnesen Haugaløkken, Martin Breivik Skaldebø and Ingrid Schjølberg. 'Monocular vision-based gripping of objects'. In: *Robotics and autonomous systems* 131 (June 2020), p. 103589.

[19] Ming En Koh, Mark Wong Kei Fong and Eddie Yin Kwee Ng. 'Aqua3DNet: Real-time 3D pose estimation of livestock in aquaculture by monocular machine vision'. In: *Aquacultural engineering* 103 (Sept. 2023).

[20] Simen Viken Grini. 'Object Detection in Maritime Environments'. Master thesis. Norwegian University of Science and Technology, Jan. 2019.

[21] M. Asada, T. Tanaka and K. Hosoda. 'Visual tracking of unknown moving object by adaptive binocular visual servoing'. In: International Conference on Multisensor Fusion and Integration for Intelligent Systems. IEEE, Aug. 1999, pp. 249–254.

[22] Thor I. Fossen. *Handbook of marine craft hydrodynamics and motion control*. 2nd ed. Hoboken, NJ: Wiley, 2021.

[23] Zongyu Chang, Yang Zhang, Zhongqiang Zheng, Lin Zhao and Kunfan Shen. 'Dynamics Simulation of Grasping Process of Underwater Vehicle-Manipulator System'. In: *Journal of marine science and engineering* 9.10 (Oct. 2021), p. 1131.

[24] Gianluca Antonelli. *Underwater Robots*. 4th ed. Cham: Springer International Publishing, 2018.

[25] Arnt Erik Stene. 'Robust Control for Articulated Intervention AUVs in the Operational Space'. Master thesis. Norwegian University of Science and Technology, June 2019.

[26] Erlend Andreas Basso. 'Dynamic Task Priority Control of Articulated Intervention AUVs using Control Lyapunov and Control Barrier Function based Quadratic Programs'. Master thesis. Norwegian University of Science and Technology, July 2019.

[27] Signe Moe. 'Guidance and Control of Robot Manipulators and Autonomous Marine Robots'. Doctoral thesis. Norwegian University of Science and Technology, Nov. 2016.

[28] J. Sverdrup-Thygeson, S. Moe, K. Y. Pettersen and J. T. Gravdahl. 'Kinematic singularity avoidance for robot manipulators using set-based manipulability tasks'. In: Conference on Control Technology and Applications. IEEE, Aug. 2017, pp. 142–149.

[29] Timothy W. McLain, Stephen M. Rock and Michael J. Lee. 'Experiments in the coordinated control of an underwater arm/vehicle system'. In: *Autonomous robots* 3.2 (Jan. 1996), pp. 213–232.

[30] T.J. Tarn and S.P. Yang. 'Modeling and control for underwater robotic manipulators - an example'. In: Proceedings of International Conference on Robotics and Automation. Vol. 3. IEEE, Apr. 1997, pp. 2166–2171.

[31] C. Canudas de Wit, E. Olguin Diaz and M. Perrier. 'Robust nonlinear control of an underwater vehicle/manipulator system with composite dynamics'. In: International Conference on Robotics and Automation. Vol. 1. ISSN: 1050-4729. IEEE, May 1998, pp. 452–457.

[32] Stian Skaalvik Sandøy. 'System Identification and State Estimation for ROV uDrone'. Master thesis. Norwegian University of Science and Technology, June 2016.

[33] Chu-Jou Wu and B Eng. '6-DoF Modelling and Control of a Remotely Operated Vehicle'. Master thesis. Flinders University, July 2018.

[34] *Hvor mye omega-3 er det i oppdrettslaks?* 4th Oct. 2021. URL: https://laksefakta.no/sunnhet-og-helse/hvor-mye-omega-3-er-det-i-oppdrettslaks/ (visited on 20/10/2023).

Appendices

Table of Contents

A Reach Alpha 5 - Kinematics

Reach System 1 Kinematic and Dynamic Properties

Blueprint Lab

Last Update: September 2019

1 Kinematic Properties

Link	d (mm)	θ	a (mm)	α
0	46.2	$\theta_0 + \pi$	20	$\pi/2$
1	0	$\theta_1 - \theta_a$	150.71	π
2	0	$\theta_2 - \theta_a$	20	$-\pi/2$
3	-180	$\theta_3 + \pi/2$	0	$\pi/2$
4	0	$-\pi/2$	0	0

Table 1: Standard DH Parameters for R5M with $\theta_a = \tan^{-1}\left(\frac{145.3}{40}\right)$

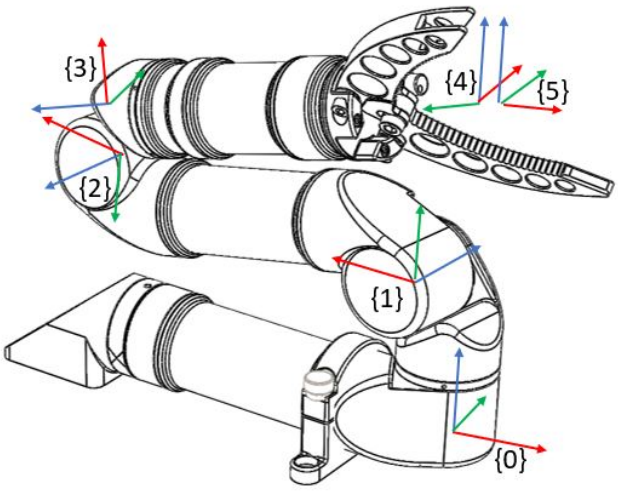


Figure 1: R5M joint frames (x,y,z)

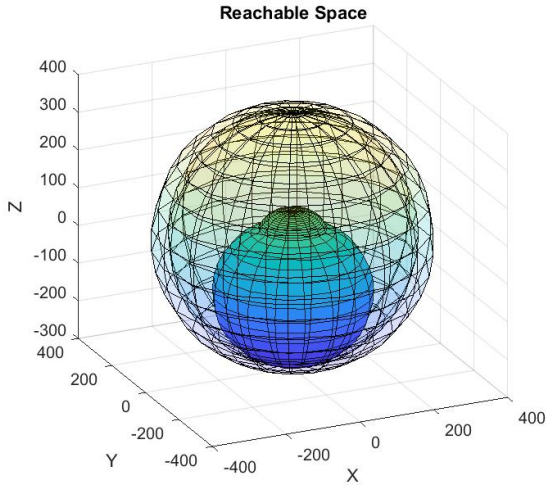


Figure 2: Reachable workspace without self collision showing inner and outer limits

1.1 Workspace

The outer reachable limits form a torus

$$(\sqrt{x^2+y^2}-a_0)^2+(z-d_0)^2 \leq (a_1+\sqrt{d_3^2+a_2^2})^2 \tag{1}$$

The inner reachable limit is the torus

$$(\sqrt{x^2+y^2}-a_0)^2+(z-d_0)^2 \geq ((39.94+a_2)^2+(145.3+d_3)^2) \tag{2}$$

The inner reachable limit with the arm in the downward position is the torus

$$(\sqrt{x^2+y^2}-a_0)^2+(z-d_0+145.3)^2 \geq (-d_3)^2 \tag{3}$$

These are shown in Figure 2.

1.2 Inverse Kinematics

From Figure 3 we can solve for the underarm solution

$$\theta_0 = tan2^1(\frac{y}{x})+\pi \tag{4}$$

$$R = \sqrt{x^2+y^2} \tag{5}$$

From Figure 4 we can solve for

$$l_1 = a_1 \tag{6}$$

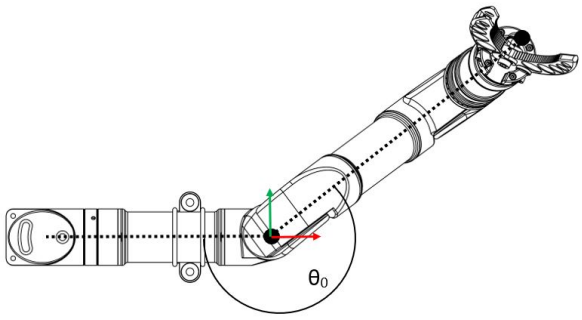


Figure 3: Calculation of θ_0

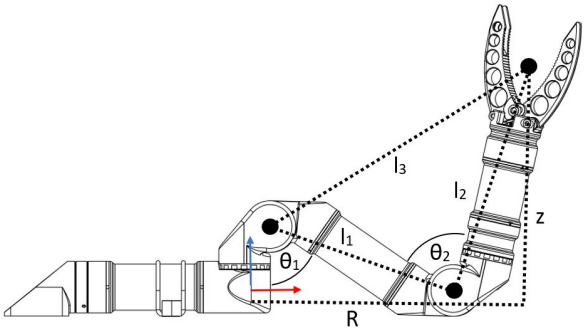


Figure 4: Calculation of θ_1 and θ_2

$$l_2 = \sqrt{a_2^2 + d_3^2} \tag{7}$$

$$l_3 = \sqrt{(R - a_0)^2 + (z - d_0)^2} \tag{8}$$

$$\theta_2 = cos^{-1}(\frac{l_1^2 + l_2^2 - l_3^2}{2l_1l_2}) - sin^{-1}(\frac{2a_2}{l_1}) - sin^{-1}(\frac{a_2}{l_2}); \tag{9}$$

$$\theta_1 = \frac{\pi}{2} + tan2^{-1}(\frac{z - d_0}{R - a_0}) - cos^{-1}(\frac{l_1^2 + l_3^2 - l_2^2}{2l_1l_3}) - sin^{-1}(\frac{2a_2}{l_1}) \tag{10}$$

Now we can also solve for the overarm solution from Figure 5

$$\theta_0 = tan2^1(\frac{y}{x}) \tag{11}$$

Finally from Figure 6

$$l_3 = \sqrt{(R + a_0)^2 + (z - d_0)^2} \tag{12}$$

$$\theta_2 = cos^{-1}(\frac{l_1^2 + l_2^2 - l_3^2}{2l_1l_2}) - sin^{-1}(\frac{2a_2}{l_1}) - sin^{-1}(\frac{a_2}{l_2}); \tag{13}$$

$$\theta_1 = \frac{3\pi}{2} - tan2^{-1}(\frac{z - d_0}{R + a_0}) - cos^{-1}(\frac{l_1^2 + l_3^2 - l_2^2}{2l_1l_3}) - sin^{-1}(\frac{2a_2}{l_1}) \tag{14}$$

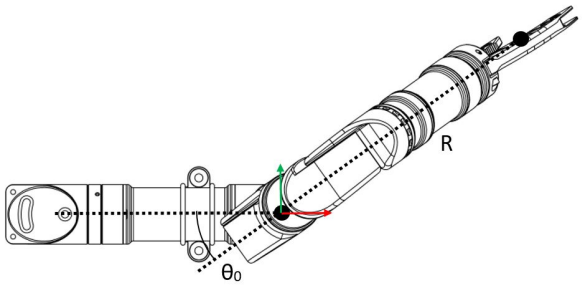


Figure 5: Calculation of θ_1 and θ_2

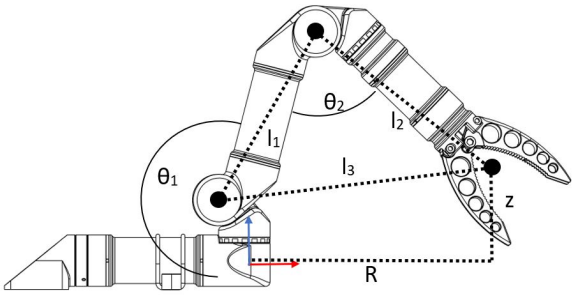


Figure 6: Calculation of θ_0

2 Inertial Properties

Link	**Mass** (kg)	**COM** (mm)	**I** $(kg.mm^2)$
0	0.341	$\begin{pmatrix} -75 & -6 & -3 \end{pmatrix}$	$\begin{pmatrix} 99 & 139 & 115 \\ 139 & 2920 & 3 \\ 115 & 3 & 2934 \end{pmatrix}$
1	0.194	$\begin{pmatrix} 5 & -1 & 16 \end{pmatrix}$	$\begin{pmatrix} 189 & 5 & 54 \\ 5 & 213 & 3 \\ 54 & 3 & 67 \end{pmatrix}$
2	0.429	$\begin{pmatrix} 73 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 87 & -76 & -10 \\ -76 & 3190 & 0 \\ -10 & 0 & 3213 \end{pmatrix}$
3	0.115	$\begin{pmatrix} 17 & -26 & -2 \end{pmatrix}$	$\begin{pmatrix} 120 & -61 & -1 \\ -61 & 62 & 0 \\ -1 & 0 & 156 \end{pmatrix}$
4	0.333	$\begin{pmatrix} 0 & 3 & -98 \end{pmatrix}$	$\begin{pmatrix} 3709 & 2 & -4 \\ 2 & 3734 & 0 \\ -4 & 0 & 79 \end{pmatrix}$

Table 2: Inertial properties for R5M

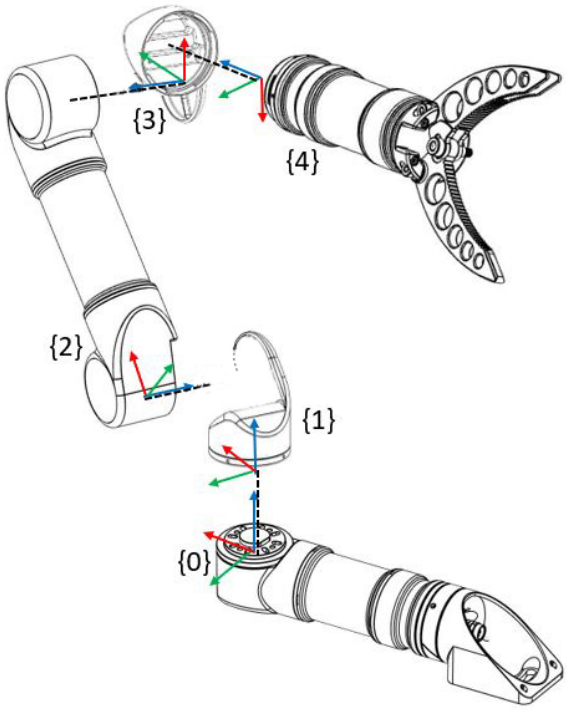


Figure 7: Inertial frames for each link (x,y,z)

3 Hydrodynamic Properties

3.1 Added Mass

We use standard equations for added mass assuming cylindrical sections with spherical ends.

Link	$(\boldsymbol{X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}})(\boldsymbol{kg})$	$(\boldsymbol{K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}})(\boldsymbol{kg.mm^2})$
0	$\begin{pmatrix} 0.017\rho & 0.189\rho & 0.189\rho \end{pmatrix}$	$\begin{pmatrix} 0 & 1414\rho & 1414\rho \end{pmatrix}$
1	$\begin{pmatrix} 0.032\rho & 0.032\rho & 0.017\rho \end{pmatrix}$	$\begin{pmatrix} 7\rho & 7\rho & 0 \end{pmatrix}$
2	$\begin{pmatrix} 0.017\rho & 0.201\rho & 0.201\rho \end{pmatrix}$	$\begin{pmatrix} 0 & 1716\rho & 1716\rho \end{pmatrix}$
3	$\begin{pmatrix} 0.032\rho & 0.017\rho & 0.032\rho \end{pmatrix}$	$\begin{pmatrix} 7\rho & 0 & 7\rho \end{pmatrix}$
4	$\begin{pmatrix} 0.226\rho & 0.226\rho & 0.017\rho \end{pmatrix}$	$\begin{pmatrix} 2443\rho & 2443\rho & 0 \end{pmatrix}$

Table 3: Added mass terms where $\rho \sim 1$ is the density in kg/L

3.2 Viscous Damping

We consider only quadratic drag assuming Reynolds numbers above the Stokes flow approximation at any significant velocity. We also consider only transla-

tional drag. We approximate the Reynolds number as

$$Re = \frac{\rho u L}{\mu} \sim \frac{10^3 * 0.1 * 0.1}{10^{-3}} \sim 10^{-4} \tag{15}$$

where ρ is the fluid density, u is the relative velocity, L is the characteristic length and μ is the dynamic viscosity, which gives a drag coefficient of $c_d \sim 0.5$. We now calculate the quadratic drag using

$$F_d = \frac{1}{2}\rho u^2 c_d A \tag{16}$$

where A is the cross sectional area.

The centre of drag is assumed to coincide with the centre of buoyancy

Link	$(\boldsymbol{X_{u\|u\|}, Y_{v\|v\|}, Z_{w\|w\|}})$ $(\boldsymbol{N/\sqrt{ms^{-1}}})$
0	$\begin{pmatrix} 0.3\rho & 1.5\rho & 1.5\rho \end{pmatrix}$
1	$\begin{pmatrix} 0.26\rho & 0.26\rho & 0.3\rho \end{pmatrix}$
2	$\begin{pmatrix} 0.3\rho & 1.6\rho & 1.6\rho \end{pmatrix}$
3	$\begin{pmatrix} 0.26\rho & 0.3\rho & 0.26\rho \end{pmatrix}$
4	$\begin{pmatrix} 1.8\rho & 1.8\rho & 0.3\rho \end{pmatrix}$

Table 4: Drag terms where $\rho \sim 1$ is the density in kg/L

3.3 Buoyancy

Link	**Volume** $(\boldsymbol{L})$	**COB** $(\boldsymbol{mm})$
0	0.202	$\begin{pmatrix} -75 & -6 & -3 \end{pmatrix}$
1	0.018	$\begin{pmatrix} -1 & -3 & 32 \end{pmatrix}$
2	0.203	$\begin{pmatrix} 73 & 0 & -2 \end{pmatrix}$
3	0.025	$\begin{pmatrix} 3 & 1 & -17 \end{pmatrix}$
4	0.155	$\begin{pmatrix} 0 & 3 & -98 \end{pmatrix}$

Table 5: Buoyancy terms with Centre of Buoyancy (COB)

4 Actuator Properties

$$torque = 90.6 * (current \pm 43.0) \tag{17}$$

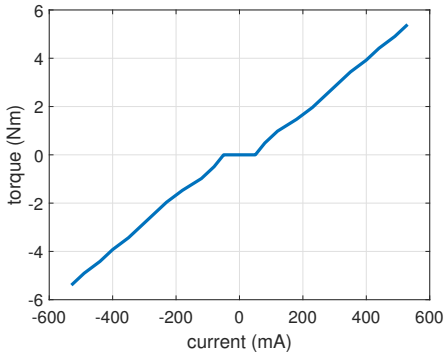


Figure 8: Plot of torque vs current for high torque joint

B BlueROV2 - Datasheet

Datasheet

Blue Robotics

BlueROV2

The World's Most Affordable High-Performance ROV

Ocean research, exploration, and adventure are all made easily accessible by our flagship product, the BlueROV2. It provides the capabilities of a high-end commercial mini-ROV at the price of the most basic commercial ROVs, making the BlueROV2 the world's most affordable inspection and research-class subsea vehicle.

The smooth, stable, and highly maneuverable ROV is comprised of six thrusters, a rugged frame, and quick-swappable batteries. Powerful but dimmable lights provide excellent illumination for the live HD video feed.

Like all Blue Robotics products, we created the BlueROV2 with high-quality parts, meticulous design, and rugged reliability with proven success in the field.

Equipped with six powerful T200 thrusters and Basic ESCs, the BlueROV2 has the best thrust-to-weight ratio in its class to perform demanding tasks. It is ideal for operations in shallow to moderate waters, with a standard 100m depth rating and up to 300m tether lengths available.

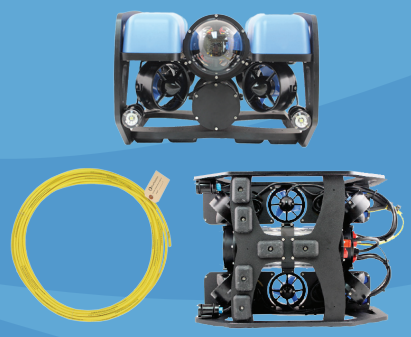
The BlueROV2 uses the open-source ArduSub software and the Navigator autopilot to provide autonomous capabilities rarely seen in mini-ROVs and hackability paralleled by none. Blue Robotics actively develops and updates its software to enhance the BlueROV2's functionality.

Your vehicle will arrive almost-ready-to-dive, with pre-built sub-assemblies and instructional materials to make the experience as straightforward and enjoyable as possible. Additional items including the topside computer, gamepad controller and batteries are not included.

At Blue Robotics, we are committed to creating quality products that are accessible to any explorer.

Product Features

- **Live 1080p HD Video (200 ms latency)**
- **Highly Maneuverable Vectored Thruster Configuration**
- **Stable and Optimized for Inspection and Research-Class Missions**
- **Easy to Use, Cross-Platform User Interface**
- **Highly Expandable with Six Free Cable Penetrators**
- **6 T200 Thrusters and Basic ESCs**
- **Standard 100m Depth Rating and Up to 300m Tether Available**
- **Battery Powered with Quick-Swappable Batteries for Long Missions**



Dare to Explore

BlueRobotics

Datasheet

Blue Robotics

BlueROV2 Technical Specifications

Revision 03/22

Physical

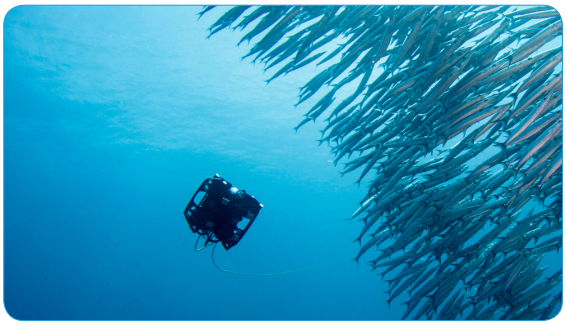
Length	457 mm	18 in
Width	338 mm	13.3 in
Height	254 mm	10 in
Weight in Air *(with Ballast and Battery)*	11-12 kg	24-27 lb
Weight in Air *(without Ballast or Battery)*	9-10 kg	20-22 lb
Payload Capacity (configuation dependent)	1.2 kg (4 x Lumens) to 1.4 kg (No Lumens)	2.6 to 3.1 lbs
Watertight Enclosure Inner Diameter	102 mm	4 in
Watertight Enclosure Inner Length	298 mm	11.75 in
Cable Penetrator Holes	18 x 10 mm	1 x 0.4 in
Buoyancy Foam	R-3318 Urethane Foam rated to 244 m	
Construction	HDPE frame, aluminum flanges/end cap, & acrylic or aluminum tubes	
Main Tube *(Electronics Enclosure)*	Blue Robotics 4 in series w/ aluminum end caps	
Battery Tube	Blue Robotics 3 in series w/ aluminum end caps	
Buoyancy Foam	R-3318 Urethane Foam rated to 244 m	
Ballast Weight	9 x 200 g stainless steel weights	
Battery Connector	XT90	

Performance

Maximum Rated Depth *(Acrylic)*	100 m	330 ft
Maximum Rated Depth *(Aluminum)*	300 m	990 ft
Maximum Forward Speed	1.5 m/s	3 knots
Thrusters	Blue Robotics T200 with WLP	
ESC	Blue Robotics Basic 30A ESC	
Thruster Configuration	6 thrusters	
	- 4 Vectored	
	- 2 Vertical	
Forward Bollard Thrust (45°)	9 kgf	19.8 lbf
Vertical Bollard Thrust	7 kgf	15.4 lbf
Lateral Bollard Thrust (45°)	9 kgf	19.8 lbf

Tether

Diameter	7.6 mm	0.30 in
Length	25-300 m	80-980 ft
Working Strength	45 kgf	100 lbf
Breaking Strength	160 kgf	350 lbf
Strength Member	Kevlar with waterblock	
Buoyancy in Freshwater	Neutral	
Buoyancy in Saltwater	Slightly Positive	
Conductors	4 twisted pairs, 26 AWG	



Lights

Brightness	2 or 4 x 1500 lumens each with dimming control
Light Beam Angle	135 degrees, with adjustable tilt

Camera

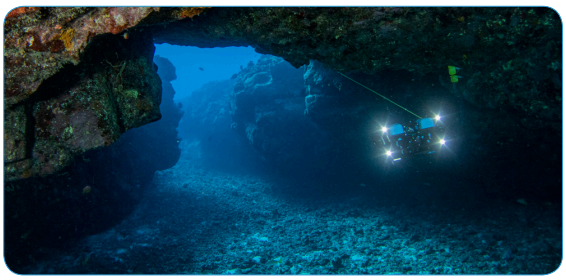
Resolution	1080p
Camera Field of View	110 degrees horizontally
Tilt Range	+/- 90 degree camera tilt *(180 total range)*
Tilt Servo	Hitec HS-5055MG

Sensors

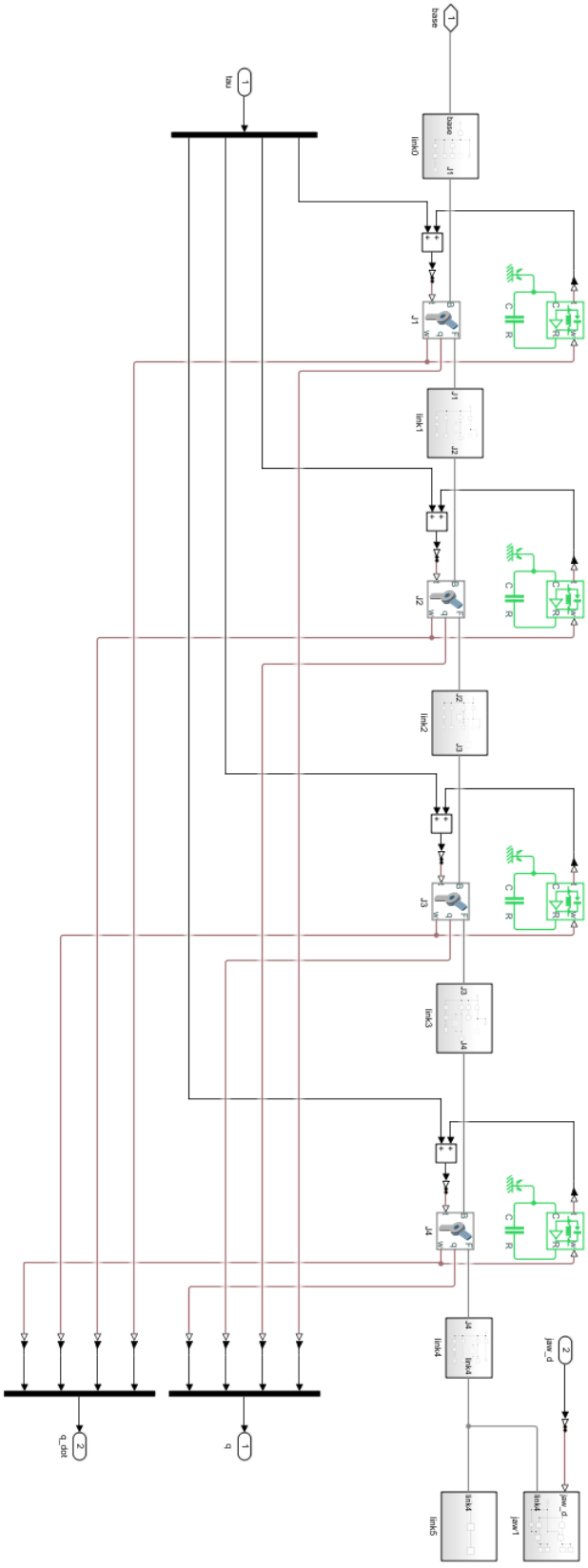
- 3-DOF Gyroscope
- 3-DOF Accelerometer
- 3-DOF Magnetometer
- Internal barometer
- Blue Robotics Bar 30 Pressure/Depth & Temperature Sensor *(external)*
- Current and Voltage Sensing
- Leak Detection

Battery *(can be changed in about 30 seconds)*

Battery Life *(Normal Use)*	2 hours w/ 18Ah battery
Battery Life *(Light Use)*	6 hours w/ 18Ah battery



C Simulink Model of the Robot Arm



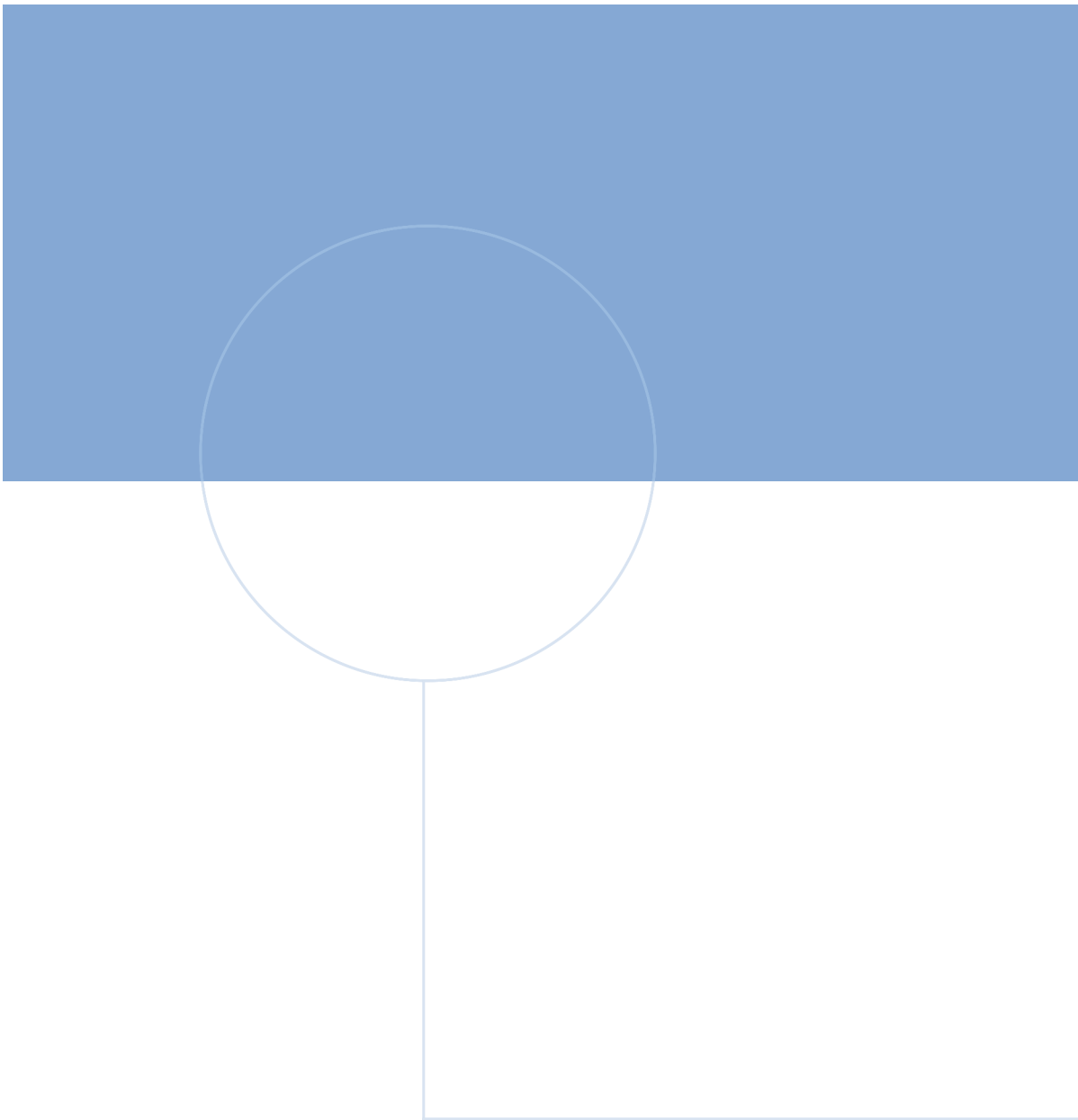
D MATLAB Code of Kinematic Control

```
1
2  function [data, jaw_d, q_dot_r, nu_r] = ...
3      kinematic_control(x_arm, x_auv, fish_pose, t)
4  %% Coefficients
5  lambda  = 0.5;      % transition between task 1 and task 2
6  lambda1 = 1;        % orientation vs position
7  lambda2 = 2;        % joint limits
8  lambda3 = 0.5;      % auv vs arm
9
10 %% Parameters
11 % Fish State
12 p_fish = fish_pose(1:3);
13 Theta_fish = fish_pose(4:6);
14 roll_fish = Theta_fish(1);
15 pitch_fish = Theta_fish(2);
16 yaw_fish = Theta_fish(3);
17 % AUV States
18 p_auv = x_auv(1:3);
19 yaw_auv = x_auv(6);
20 % other
21 robot_arm_placement = robot_arm_placement_func();
22
23 %% Rotation Matrices
24 % Robot arm placement
25 R_base_to_auv = rot_mat(robot_arm_placement(4),0,0);
26 T_base_to_auv = robot_arm_placement(1:3);
27 H_base_to_auv = [R_base_to_auv, T_base_to_auv;
28                  zeros(1,3),           1];
29 % Transform from auv frame to world frame
30 R_auv_to_NED = rot_mat(x_auv(4),x_auv(5),x_auv(6));
31 T_auv_to_NED = [x_auv(1); x_auv(2); x_auv(3)];
32 H_auv_to_NED = [R_auv_to_NED, T_auv_to_NED;
33                  zeros(1,3),          1];
34
35 H_base_to_NED = H_auv_to_NED * H_base_to_auv;
36 R_base_to_NED = H_base_to_NED(1:3,1:3);
37
38 R_fish_to_NED = rot_mat(roll_fish, pitch_fish, yaw_fish);
39 T_fish_to_NED = p_fish;
40 H_fish_to_NED = [R_fish_to_NED, T_fish_to_NED;
41                  zeros(1,3),          1];
42
43 %% End-Effector pose in world frame
44 [p_ee_base, R_ee_to_base] = forward_kinematics(x_arm(1:4));
45 p_ee = H_base_to_NED * [p_ee_base(1:3);1]; p_ee = p_ee(1:3);
46 R_ee_to_NED = R_base_to_NED * R_ee_to_base;
47 ZYX_ee = rotm2eul(R_ee_to_NED);
48 ZYX_ee(3) = mod( ZYX_ee(3) + pi/2, 2 * pi/2 ) - pi/2;
49 eul_ee = [ZYX_ee(3);ZYX_ee(2);ZYX_ee(1)];
50 sigma = [p_ee; eul_ee];
51
52 %% Compute Jacobian
53 p_ee_auv = H_base_to_auv*[p_ee_base;1]; p_ee_auv = p_ee_auv(1:3);
54 J_arm_base = compute_jacobian(x_arm(1:4));
55 J = compute_system_jacobian(J_arm_base, R_auv_to_NED, ...
```

```
                                R_base_to_NED, p_ee_auv);

%% Control Law
%%%%%%%%%% TASK 1: positioning auv and keep fish in FOV %%%%%%%%%%
J1 = [eye(3), zeros(3,1), zeros(3,4);
       zeros(1,3),     1,      zeros(1,4)];
J1_T = J1';

vec_auv_to_fish = p_fish - p_auv;
y_axis_fish = R_fish_to_NED(:,2);
pos_e = vec_auv_to_fish - dot(vec_auv_to_fish, y_axis_fish) *
    y_axis_fish;

yaw_FOV_d = atan2(vec_auv_to_fish(2), vec_auv_to_fish(1));
yaw_FOV_e = yaw_FOV_d - yaw_auv;

sigma1_tilde = [pos_e; yaw_FOV_e];
K_1 = [0.1*ones(3,1); 1];
e0 = 0.05;
Lambda = sigmoid(sigma1_tilde, e0, lambda);

zeta1_r = J1_T*(K_1.*sigma1_tilde);

%%%%%%%%%% TASK 2: convergence %%%%%%%%%%
pos_d = p_fish;
roll_d = -pitch_fish; % roll_d = mod( roll_d + pi/2, 2 * pi/2 ) -
    pi/2;
pitch_d = -asin((p_fish(3)-p_ee(3)) / norm(p_fish - p_ee));
yaw_d = yaw_fish + pi/2; yaw_d = mod( yaw_d + pi/2, 2 * pi/2 ) -
    pi/2;
sigma_d = [pos_d;roll_d;pitch_d;yaw_d];

sigma2_tilde = sigma_d - sigma;
% pitch error becomes very larg when position error is small
Lambda_pitch = sigmoid(sigma2_tilde, 0.05, 0.2);
sigma2_tilde(5) = Lambda_pitch*sigma2_tilde(5);
if norm(sigma2_tilde(1:3)) < 0.01
    sigma2_tilde(5) = 0;
end

k1 = [2, 2, 2, 1, 1, 1]; % general weighting
k2 = orientation_vs_position(sigma2_tilde, lambda1);

K1 = k1(1)*k2(1);
K2 = k1(2)*k2(2);
K3 = k1(3)*k2(3);
K4 = k1(4)*k2(4);
K5 = k1(5)*k2(5);
K6 = k1(6)*k2(6);

K_2 = [K1;K2;K3;K4;K5;K6];

w1 = [15,15,15,15,1,1,1,1]; % general weighting
w2 = joint_limits(x_arm(1:4), x_arm(5:8), lambda2);
w3 = auv_vs_arm(norm(sigma2_tilde(1:3)), lambda3);

W1 = w1(1)*w2(1)*w3(1);
W2 = w1(2)*w2(2)*w3(2);
```

```
W3 = w1(3)*w2(3)*w3(3);
W4 = w1(4)*w2(4)*w3(4);
W5 = w1(5)*w2(5)*w3(5);
W6 = w1(6)*w2(6)*w3(6);
W7 = w1(7)*w2(7)*w3(7);
W8 = w1(8)*w2(8)*w3(8);

W_2 = diag([W1,W2,W3,W4,W5,W6,W7,W8]);

J2_T = W_2\J';

zeta2_r = J2_T*(K_2.*sigma2_tilde);

%%%%%%%%%% TASK 3: manipulability %%%%%%%%%%
[zeta3_r, w] = manipulability_optimization(x_arm(1:4));

%%%%%%%%%% Control Law %%%%%%%%%%
nu_r = Lambda*zeta1_r(1:4) + (1-Lambda)*zeta2_r(1:4);
q_dot_r = zeta1_r(5:8) + zeta2_r(5:8) + zeta3_r(5:8);
jaw_d = deg2rad(30);

%% Velocity Limit
for i = 1:4
    if abs(q_dot_r(i)) > 0.1
        q_dot_r(i) = sign(q_dot_r(i))*0.1;
    end
    if abs(nu_r(i)) > 0.2
        nu_r(i) = sign(nu_r(i))*0.2;
    end
end

%% Initial States
if t < 10
    nu_r = zeros(4,1);
    q_dot_r = zeros(4,1);
end

%% Data for plotting
data = [sigma; sigma_d; sigma2_tilde; w; t];
```

Listing 5: Kinematic Control



NTNU

Norwegian University of
Science and Technology