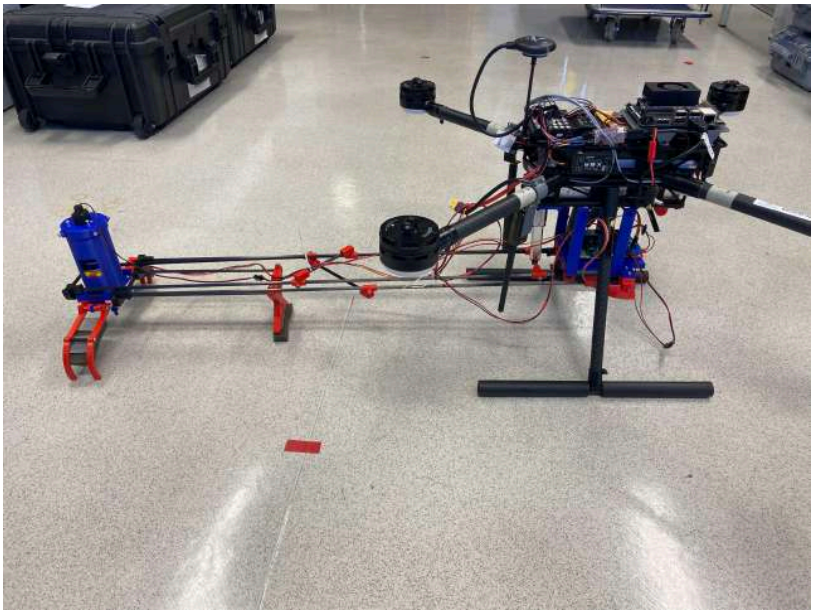
Master of Sciences in Robotics

Master's Thesis - Project capture

Fast grasping mechanism for aerial and marine drone automatic capture



Student:
Arthur Dietrich

Supervisors:
Prof. Rita Cunha
Prof. Bruno Guerreiro
Prof. Auke Ijspeert
Prof. Pierre-Alain Mäusli

August 31, 2023

Acknowledgement

This thesis is the culmination of many years of study. I would like to express my heartfelt gratitude to my family for their unwavering support and to my friends for their constant presence. A special thanks goes to Jules for his courage and inspiration, and to Olivier for being the incredible friend and person he is.

I extend my sincere appreciation to Proj. Auke Ijspeert for agreeing to supervise my thesis. My deepest thanks also go to Prof. Rita Cunha, Prof. Bruno Guerreiro, and Prof. Pierre-Alain Mäusli for their invaluable guidance and support throughout this journey. This thesis would not have been possible without their supervision and insightful ideas. I am also grateful to Marcelo Jacinto, Francisco Velez, and Diogo Oliveira for generously dedicating their time to help me understand new concepts during this research.

My gratitude goes to Mr. Tojeira and Mr. Rufino for their assistance in building the prototype. Lastly, I want to express my heartfelt appreciation to everyone in the DSOR Lab. Special thanks to Marcelo, João, Pedro, Gil, João, José, Pedro, and Francisco for their warm welcome and making me feel like an integral part of the team.

I am pleased to acknowledge that this thesis was partially supported by the Swiss European Mobility Program and the Portuguese Foundation for Science and Technology (FCT/MCTES) through projects CAPTURE (PTDC/EEI-AUT /1732/2020), CTS (UIDB/EEA/00066/2020), and LARSYS (UIDB/EEA/50009/2020).

Abstract

This thesis addresses the design of a rapid grasping mechanism for the automated capture of aerial and marine drones. The proposed solution involves an articulated robotic arm equipped with a gripper mounted on a quadcopter. This quadcopter tracks the position of a target drone by employing an extended Kalman filter to fuse measurements.

The design of the gripper mechanism entails the creation of a passively actuated gripper triggered through contact with the target drone. Following an in-depth examination of the gripper's mechanical aspects, a final design is proposed, and a prototype is fabricated using 3D printing technology. Engineered to be activated by a customizable force, the gripper effectively captures drones of known shapes due to its adaptable finger design. The articulated robotic arm facilitates a capture process that is free from the downwash effects generated by the propellers of the shuttle drone.

The localization algorithm harnesses the power of an extended Kalman filter to integrate measurements from various sources. These measurements are currently obtained via the ArUco fiducial marker system for close-range tracking, depth measurements facilitated by a stereo camera, and GPS positioning. Although the system has not yet been subjected to real flight testing, the preliminary indoor results are promising. These findings highlight accurate tracking capabilities and efficient capture potential.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

In recent years, Unmanned Aerial Vehicles (UAVs) have emerged as pivotal tools with the potential to revolutionize various sectors, including logistics, agriculture, surveillance, and more. The adaptability and flexibility of UAVs have given rise to an array of applications, prompting the exploration of complex challenges that must be addressed to fully harness their capabilities.

One of the significant challenges arises from the need to design specialized UAVs tailored to specific tasks. For instance, there is a demand for fixed-wing drones with extended endurance for tasks such as long-range reconnaissance or mapping missions. Simultaneously, the requirement for vertical take-off and landing (VTOL) capabilities is vital to ensure accessibility and maneuverability in diverse environments. Finding the right balance between endurance and VTOL capabilities is crucial to optimizing the efficiency of these vehicles.

Furthermore, the advent of UAVs has led to scenarios where these unmanned systems need to interact with other vehicles or objects in dynamic and sometimes non-cooperative environments. This has brought about the need for innovative security measures to counteract potential threats. The challenge lies in developing solutions that enable the safe and effective removal of drones or objects from restricted areas, even when these entities actively evade capture. Tackling this challenge involves intricate coordination, advanced control strategies, and efficient estimation techniques to ensure successful interaction between UAVs and their surroundings.

As UAV technology continues to evolve, addressing these challenges becomes increasingly paramount. By delving into the intricacies of designing specialized vehicles and devising effective security measures, the potential of UAVs to shape various industries can be fully realized. This motivation propels researchers and engineers to explore innovative solutions that not only elevate the capabilities of UAVs but also contribute to the broader technological landscape.

1.2 Objectives and Problem Definition

The *CAPTURE* project [1] encapsulates a set of overarching objectives and challenges aimed at enhancing the effectiveness of shuttle drone operations. These objectives encompass a range of intricate tasks and multifaceted scenarios, including:

- **Optimal and Cooperative Trajectory Planning**: The project aims to devise innovative methodologies for orchestrating trajectories that ensure optimal coordination among a diverse ensemble of vehicles, catering to their unique characteristics and roles within the operation.

- **Cooperative, Hybrid, and Distributed Control**: To enable flawless rendezvous maneuvers, the project focuses on formulating control strategies that seamlessly blend cooperative, hybrid, and distributed approaches. These strategies are designed to navigate intricate and dynamic aerial environments, ensuring successful interactions between the shuttle drones and the vehicles they are launching or capturing.
- **Cooperative and Distributed Estimation**: The project addresses the challenge of accurately estimating the motion of multiple entities in a cooperative and distributed manner. Precise estimation of the positions and orientations of shuttle drones, other vehicles, and the surrounding environment is pivotal for ensuring the reliability and success of launch and capture operations.
- **Non-Cooperative Strategies and Differential Games**: In scenarios where cooperation may not be guaranteed, the project delves into non-cooperative strategies. This entails the formulation of sophisticated estimation, control, and planning techniques rooted in differential games theory. These strategies empower the shuttle drones to effectively capture non-cooperative objects or drones that may actively resist capture attempts.

The project distinguishes between two fundamental scenarios:

1. **Cooperative Scenario**: This scenario involves shuttle drones collaborating with vehicles destined for launch or capture (**Figure 1**). Whether relaying crucial motion information or synchronizing movements, the shuttle drones play a facilitative role in ensuring successful maneuvers.
2. **Non-Cooperative Scenario**: In this scenario, the project tackles the launch and capture of objects or drones that might not actively cooperate with the shuttle drones. These objects or drones may exhibit passive or active resistance to capture attempts, necessitating specialized strategies to overcome their non-cooperative behavior.

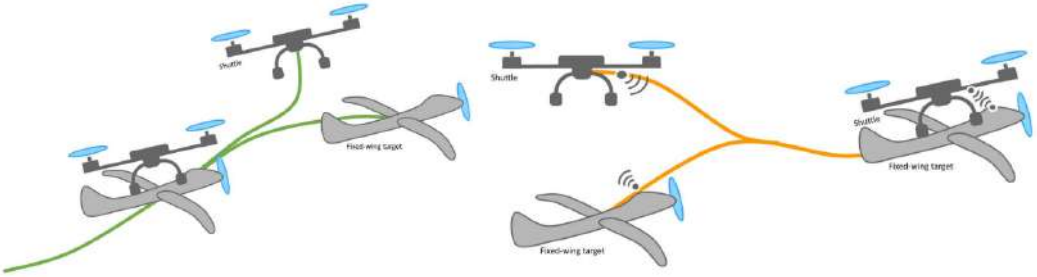


Figure 1: Basic diagrams for launch and capture maneuvers.

Problem Definition:

In the context of the *CAPTURE* project, the primary problems to be addressed include:

- **Localization of Target Drones (TD)**: Develop innovative solutions for accurately localizing the target drones with respect to the shuttle drones, enabling precise positioning information crucial for successful launch and capture operations.
- **Capture Mechanism Design**: Create robust and effective mechanisms to capture the target drones, ensuring a secure and stable grip during both cooperative and non-cooperative capture scenarios.

To capture the TD, various mechanisms are being considered. Among these, two solutions are particularly viable. The first one is a straightforward approach that employs a hook, illustrated in **Figure 2**.

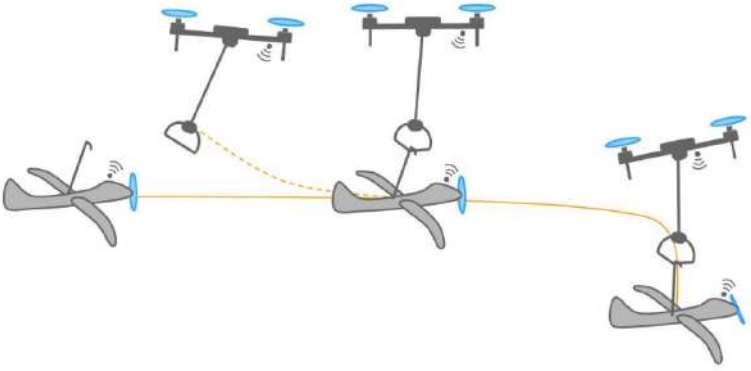


Figure 2: Hook mechanism to catch the TD.

The second option is more intricate and involves a gripper that can secure either the body or the wings of the TD. To mitigate disturbances caused by the downwash from the SD's propellers, the gripper needs to be positioned away from the propellers, as shown in **Figure 3**.

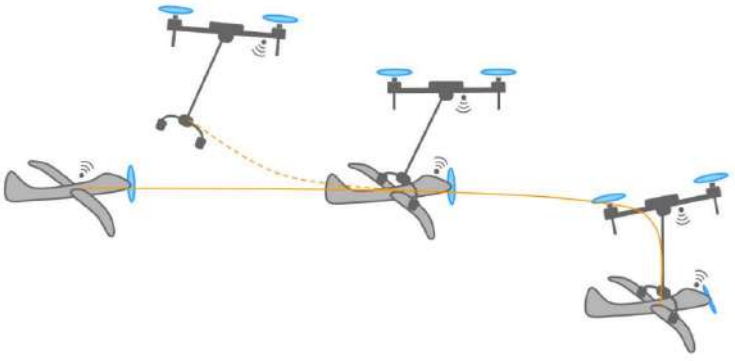


Figure 3: Deported gripper to catch the TD.

Requirements[guerreiro_capture_nodate]:

The selection of both the TD and SD, as well as the design of the catching mechanism, must adhere to specific requirements[1][2](**Tables 1, 2, 3 and 4**).

ReqID	Description	Performance
RTa1a	Weigh less than 1kg MTOW	$\pm 100g$
RTa1b	Weigh less than 2kg MTOW	$\pm 200g$
RTa2	Minimum cruise speed below 50km/h	$+5km/h$
RTa3	Minimum endurance of 30 min	$\pm$10 min
RTa4	Exchange information with shuttle drone (in cooperative maneuvres)	Minimum rate of 10 Mbps and maximum delay of 20 ms
RTa5	Exchange information with ground infrastructure	Minimum rate of 200 kbps and maximum delay of 1s

Table 1: Target functional requirements and associated performance.

ReqID	Description	Performance
RSh1	Be able to launch and capture the target drone	80% success rate
RSh2	Have sensors to measure local pose of targets	1 cm STD1 in position and 3 deg STD in attitude
RSh3	Have on-board processing capability for optimization and computer vision tasks	CPU >50 GFLOPS, GPU >1 TFLOPS2 RAM $\geq$ 8GB
RSh4	Exchange information with targets (in cooperative maneuvers)	Minimum rate of 10 Mbps and maximum delay of 20 ms
RSh5	Exchange information with ground infrastructure	Minimum rate of 200 kbps and maximum delay of 1 s
RSh6	Match the minimum forward speed of target vehicle (related with RTa1)	$\pm$10%
RSh7	Minimum endurance of 15 min	$\pm 5min$
RSh8	Avionics, mechanism, and target drone must be below 90% of drone MTOW	$\pm$5%

Table 2: Shuttle drone functional requirements and associated performance

[1]Note that RTa1a and RTa1b represent two alternatives of the same requirement RTa1, for which different vehicles and options will have to be considered.

[2]Note that MTOW means Maximum Take-Off Weight.

ReqID	Description	Performance
RMe1	Weight less than 400g	$\pm 100g$
RMe2	Reusable for a sequence of launch and capture maneuvers more than 10 times	± 2
RMe3	Residual energy spent while not in a launch/capture maneuver	
RMe4	Sustain impact from target drones	
RMe5	The terminal part must be water resistant	

Table 3: Capture/launch mechanism functional requirements and associated performance

ReqID	Description
RSa1	Each drone must have a safety pilot ready to intervene
RSa2	Drones must have in place failsafe measures and procedures
RSa3	Operation should occur in places without external intervenients
RSa4	Pilots and 3rd party liability must be insured

Table 4: Safety requirements

1.3 Proposed Solutions

This thesis will focus on designing and prototyping a lightweight and automated grasping system intended for use by aerial drones during capture maneuvers involving small, lightweight aerial drones or marine surface vessels. In an initial approach, assuming the target drones will be cooperative, either by relaying their state or being equipped with known elements, a sensor system equipped with a camera and possibly a distance sensor will assist both the drones and their grasping mechanisms in determining the appropriate moment to initiate grabbing or release actions.

The TD investigated in this thesis is the Multiplex BK EasyGlider 4 [2] (**Figure 4**), with its specifications outlined in **Table 5**.

Model	**Total Weight [kg]**	**Wingspan [m]**	**Speed [km/h]**	**Based Price []**
Multiplex BK EasyGlider 4	1.1	1.8	NA	225

Table 5: TD characteristics.

Regarding the SD, we will employ the quadcopter T-Motor M690B[3] (**Figure 4**) and assume that it can achieve a matching cruise speed with the TD.

(a) Multiplex BK EasyGlider 4.

(b) T-Motor M690B.

Figure 4: TD and SD used in this project.

While focusing on aerial vehicles as the TD, we will not overlook marine vehicles and aim to develop a versatile and easily adaptable solution.

We will also concentrate on the second catching method illustrated in **Figure 3**, leaving the hook mechanism for other studies.

1.3.1 Catching Scenarios

In the context of capturing a target drone (TD) using a gripper mounted beneath a quadcopter (shuttle drone or SD), an essential aspect is determining the optimal strategy for grasping the TD effectively and securely. This choice of strategy involves various factors such as the specific location to grasp the TD, whether it be the body or the wings, as well as the positioning of the gripper in relation to the wings. Furthermore, considerations include the use of a single gripper or multiple grippers, and the potential incorporation of differential velocities between the TD and SD. Selecting the appropriate strategy is crucial for achieving reliable and efficient automatic capture, minimizing potential risks, and ensuring the success of the overall system. In this section, we delve into an in-depth exploration of the catching scenarios to evaluate and determine the most suitable approach.

The primary distinction that can be made lies in the choice of the catching point on the aircraft. Two potential solutions are under consideration: the wings or the body of the TD. Grasping the wings of the TD appears to be the intuitive option, offering improved stability due to the symmetry and two catching points provided by the wings. Moreover, as the center of mass (CoM) aligns with the wings along the longitudinal axis, capturing the wings can maintain the plane in a horizontal position after the catch. However, this approach requires two grippers, thereby increasing the overall mechanism's mass. Additionally, the non-cylindrical shape of the wings necessitates more complex grippers, such as tendon-driven graspers, to securely wrap around them. This poses the challenge of target specificity, making it difficult to adapt the gripper from one TD to another, especially in the case of marine TDs that lack wings. Alternatively, capturing the aircraft by its body offers simplicity, lightweight design, and adaptability to different TDs. Utilizing rigid fingers suitable for the cylindrical body allows for a range of finger designs tailored to each TD. However, the primary drawback of grasping the body is the previously mentioned issue of the CoM placement at the wings level, which can cause the plane to tilt forward if caught behind the wings or backward if caught in front. This challenge can be addressed by employing contact points along the longitudinal axis on the top of the TD. Lastly, grabbing the TD behind the wings ensures enhanced stability, as the airflow will naturally push the wings against the gripper, facilitating a firmer grip.

The advantages and disadvantages associated with these two options are summarized in **Table 6**.

	Wings	**Body**
Pros	- More stable (2 catching points) - CoM directly caught	- Light and simple - Easily adaptable to different targets
Cons	- Too target-specific - Heavy and complex	- Potential stability issues due to the CoM placement

Table 6: Advantages and disadvantages of the grabbing location on the TD.

The second distinction in the catching strategy involves the disparity in velocities between the two drones. Assuming that the SD can match the speed of the TD, two scenarios arise. In the first scenario, where one of the two drones flies faster than the other, the contact between them occurs horizontally. This approach ensures that the trigger force applied during the catch does not significantly affect the behavior of the TD. However, it limits the attempt to a single try, meaning that if the catch is missed, the maneuver must be restarted from the beginning. In the second scenario, when both drones have the same speed, the contact is made vertically. While a large trigger force can potentially disrupt the behavior of the TD in this case, multiple attempts to catch the TD are possible. Furthermore, the contact area is larger, reducing the need for high accuracy during the catch.

Based on the considerations discussed, the chosen strategy entails capturing the target drone by its body, behind the wings, utilizing contact points to maintain a horizontal orientation, and applying a vertical force to activate the gripper. This approach is preferred due to its simplicity, adaptability, enhanced stability, and the opportunity for multiple catch attempts, making it a suitable choice for the successful capture of the target drone.

1.4 Thesis Outlines

This thesis will be organized as follows: Chapter 2 briefly examines related research to provide insights into existing solutions. Chapter 3 presents the mechanical design of the gripper, the robotic arm, and the electronics, offering both theoretical considerations and results analysis. In Chapter 4, localization algorithms and the filtering method used to merge measurements will be discussed, including both theoretical aspects and performance evaluations. Lastly, Chapter 5 concludes the thesis by assessing the fulfillment of requirements and briefly addressing limitations and potential future work.

2 Related Work

Before delving into the intricacies of this thesis, it is imperative to thoroughly examine the pre-existing solutions that have been developed to address our study problem. For the sake of clarity, this related work section will be divided into two parts. First, we will scrutinize the existing grippers, followed by an analysis of the target localization solutions that are already in existence.

2.1 Grippers

In the realm of drone capture mechanisms, various solutions have been explored, including those utilizing hooks [4] or nets [5]. However, the primary focus of this study centers on the design and implementation of a gripper, as detailed in the introduction. Grippers employed in aerial domains predominantly adopt a passive actuation approach due to payload weight constraints. Active closure of the gripper would necessitate heavy motors for swift gripping, which is not ideal for lightweight aerial vehicles.

Within the domain of gripper mechanisms, a distinction can be drawn based on the method of triggering. The first category comprises grippers that utilize passive components, often involving springs, loaded and held in position by an end-stop. Mechanical contact displaces the end-stop, releasing the spring and causing the gripper's fingers to close rapidly. The speed of closure is influenced by the pressure exerted by the compressed spring, affecting both the closing speed and the force required to trigger the mechanism.

Examples of this design approach can be observed in Stewart et al. [6], where a gripper mounted beneath an RC plane emulates the behavior of a bird of prey. The gripper triggers upon contact, boasting lightweight construction and quick operation, suitable for smaller targets weighing around 30 grams. Another illustration is presented by Chen et al. [**chen**], featuring a gripper activated by contact and actuated using loaded tendons and springs. This design pathway naturally leads to the consideration of the second gripper category, which employs an active release system.

A notable instance of this second category is the gripper developed by McLaren et al. [7]. Employing tendon-driven mechanics, the gripper's fingers are mounted on torsion springs, and tension is created via tendons winding around a motor axis. A distinctive aspect is the integration of a dog collar system to engage or disengage the motor from the axis of rotation. This approach enables increased spring preloading, optimizing the trigger mechanism by relying on sensors rather than contact or friction. While enhancing closing time, this system introduces complexity and additional weight due to the need for an extra actuator to release the mechanism.

The aerial domain has also witnessed the development of other grippers, designed for perching

[8][9] and object capture [10], showcasing the diverse range of gripper applications.

2.2 Target Localization

Moving on to the realm of target localization, it's noteworthy that most techniques for target detection rely heavily on computer vision. In the aerial domain, the more advanced approaches focus on detecting and grasping objects with unknown shapes. This holds true for the study by Thomas et al.[11] where a monocular camera is employed to identify objects suitable for grasping or perching, and also for the work of Li et al.[12] who devised a network to determine optimal ways of grabbing an object.

In the context of this study, our advantage lies in the fact that we possess knowledge about the shape of the Target Drone. Since we are approaching this cooperatively, we can incorporate recognition patterns onto the target drone to aid in its detection.

This leads us to the utilization of fiducial markers known as ArUco markers. These markers are widely used for pose estimation due to their precision in both orientation and position. They were employed during Francisco Azevedo's master's thesis[**azevedo_francisco_nodate**] at the ISR laboratory[13], as part of the REPLACE project[14], to detect the position of a package. Combining multiple markers in a board configuration offers reliable pose estimation from various angles, enabling the grasping and transportation of the package by a drone. However, this technique is most suitable for close-range detection.

For detecting objects like our Target Drone from a more substantial distance, we can explore the paper by Kharchenko et al.[15] Here, the popular algorithms YOLO v3 and tiny YOLO v3 are assessed for their ability to detect planes on the ground from satellite images. YOLO (You Only Look Once) is an object detection algorithm capable of identifying and locating multiple objects within images or video frames. Designed for real-time object detection in embedded systems, its application could be extended to our scenario since the planes on the ground share a similar shape to our target drone. However, it's important to note that training this algorithm requires a substantial amount of labeled data, a resource we currently lack. Additional computer vision techniques are employed to track objects such as balloons, as demonstrated in the work by Garcia et al.[16]

One fundamental principle we can ascertain is the necessity for multiple sources of measurement. To fuse these various measurements, solutions like the Kalman filter are employed. In the study by Stovner et al.[17], a novel attitude estimator called the multiplicative exogenous Kalman filter was introduced. This estimator marries the stability properties of a nonlinear observer with the near-optimal steady-state performance of the linearized Kalman filter, making it a valuable tool for estimation in nonlinear systems.

3 Mechatronic Design

In this chapter, we delve into the mechatronic design of the capturing mechanism, which comprises the gripper itself and its integration with the SD through a robotic actuated arm. By examining the intricate details of this design, we aim to highlight the key components and considerations that contribute to the overall functionality and efficiency of the capturing system.

3.1 Methods

In this section, we delve into the methods employed to develop our mechatronic design, a crucial component of our project. We discuss the iterative process of designing various elements, such as the gripper, the robotic arm, and the associated electronics. Our focus lies in achieving robustness, reliability, and efficiency in these components.

3.1.1 Gripper

In line with the specified requirements, the gripper needs to adhere to certain criteria. It should be lightweight, considering that the mass of the mechanism is limited by the maximum take-off weight (MTOW) of the SD. Compactness is crucial for efficient integration and maneuverability. Furthermore, the gripper should offer easy triggering, fast closure, and convenient reloading/reopening (without necessarily requiring fast reopening). Additionally, it must possess a holding force that is substantial enough to securely retain the plane, ensuring stable transportation. These defined requirements shape the mechanical design considerations of the gripper, enabling it to meet the objectives of the capturing system effectively.

In determining the closing mechanism for the gripper, two options present themselves: passive force utilizing a spring, or active force employing a motor. Typically, using a passive spring results in faster closure compared to using a motor. However, when using a motor, the force applied to hold the TD can be actively adjusted, while for a passive actuation, it relies on the characteristics of the spring. Additionally, the motor can serve the dual purpose of reloading the mechanism, whereas an independent system is required in the case of a spring. A notable advantage of passive closure is its lack of dependency on sensors or electronics to trigger the closing action, as it occurs instantaneously upon contact. Considering these pros and cons and drawing inspiration from the current state-of-the-art, we opt to utilize a spring for closing the gripper. Furthermore, a compression spring proves more suitable than an extension spring, as it allows for greater compactness while delivering comparable performance.

To facilitate a structured discussion, the remaining part of this subsection will be divided into three sections: gripper operation, loading mechanism, and component sizing.

- **Gripper operation**

  The underlying concept for the gripper design draws inspiration from the paper "*How to Swoop and Grasp Like a Bird With a Passive Claw for High-Speed Grasping*"[6]. In this

approach, a spring is preloaded and held in place by a specific component (blue part in **Figure 5**). When triggered by an external force, this component releases the spring, which then applies pressure to close the claws via a set of interconnected arms. This design leverages the mechanical properties of the spring and the coordinated movement of the arms to achieve a swift and efficient gripping action. To ensure a consistent pressure on the TD once the gripper is closed, the spring remains compressed at the end of the gripping movement.

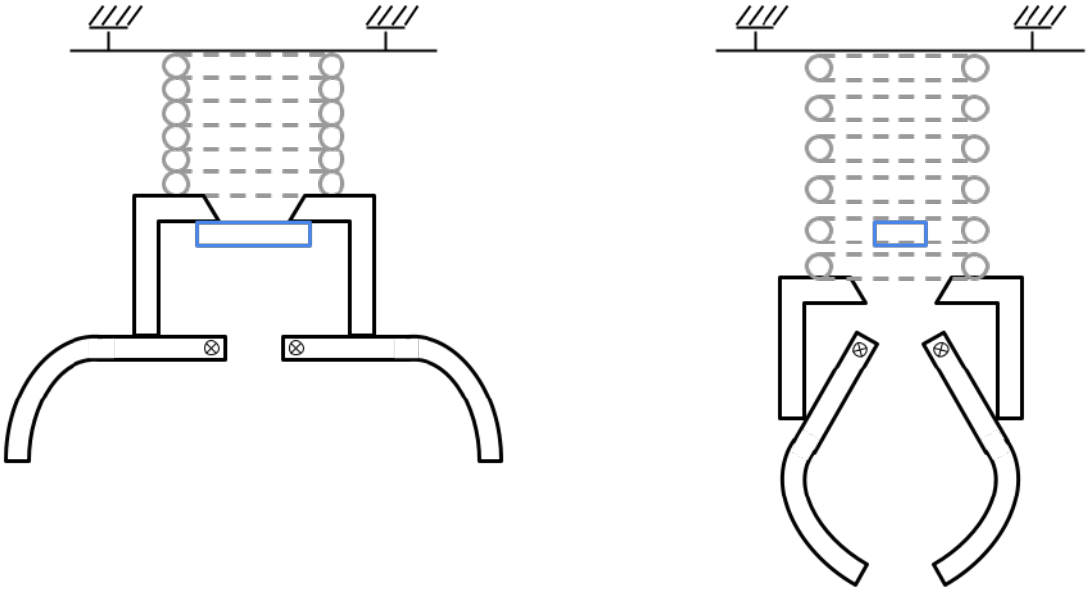


Figure 5: Open and closed states of the gripper.

To address the delicate nature of the blue part responsible for holding and releasing the spring, a design solution has been developed to ensure both ease of release and a secure open state. Initially, a pivot-mounted part triggered by a vertical movement was considered, but this configuration posed challenges. With this setup, the spring would only be held at a small portion of its perimeter, making it overly sensitive and susceptible to triggering, especially in the presence of vibrations. To overcome this limitation, an alternative design has been devised:

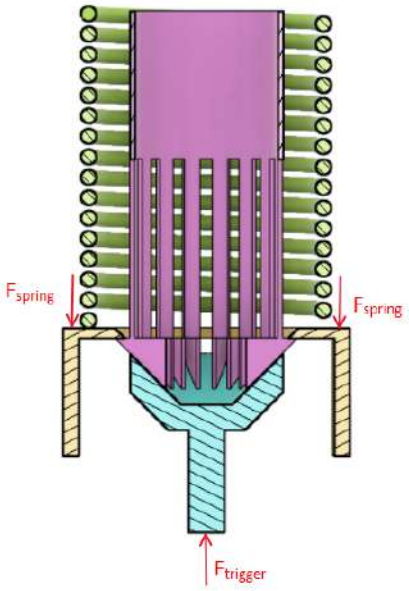


Figure 6: Trigger mechanism consisting of two interlocking conical parts.

In **Figure 6**, the centerpiece of the mechanism features a cylinder with a conical head, incorporating slots that enable retraction when subjected to pressure from another conical part, specifically the bottom component. The trigger force is generated by the collision of the mechanism's trigger rod with the body of the TD. This configuration ensures that the spring remains compressed along its entire perimeter, enhancing its resistance to vibrations. Additionally, the hollow nature of the centerpiece facilitates the seamless integration of the reloading system. By modifying parameters such as the cone angle, number and size of the slots, or the material composition, the friction and consequently the release force can be adjusted. However, it is important to note that these parameters cannot be modified once the mechanism is assembled. Thus, there is a need for a manual and easily adjustable method to fine-tune the required trigger force.

The most straightforward parameter to adjust is the friction. It's important to note that the friction force is proportional to the normal force which is proportional to the spring force :

$$F_{friction} = \mu_s F_N \propto F_{spring} \tag{1}$$

To enhance the friction force, we can elevate the spring force, resulting in a higher force needed to activate the mechanism. One approach to achieve this is by compressing the spring further, which can be accomplished by incorporating screws at the top of the mechanism as illustrated in **Figure 7**. A drawback of this method is the necessity to

adjust multiple screws in unison to fine-tune the triggering force. Nonetheless, due to its straightforward nature, we opted to retain this solution.

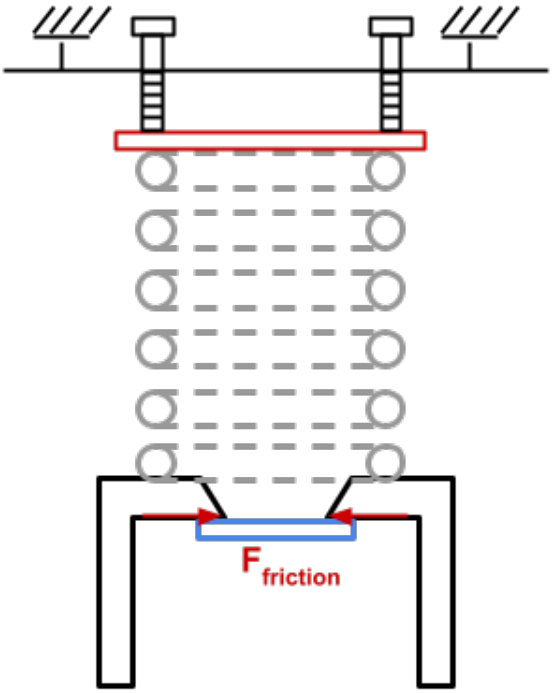


Figure 7: Screws added to tune the trigger sensitivity.

- **Loading mechanism**

  When developing the reloading system, the primary objective is to ensure ease of deployment and high efficiency without compromising the closing time of the mechanism. Several potential solutions have been evaluated, with one particularly suitable option being to position the motor on the drone itself. This arrangement minimizes inertia and improves overall system dynamics. However, this configuration presents a challenge regarding the cable or system operated by the motor for recharging the mechanism. It necessitates a complex geometry (see **Figure 8**) to enable the spring to be recharged effectively, regardless of the angle of the robotic arm. Crucially, the loading force must always remain parallel to the axis of the spring to guarantee proper reloading.

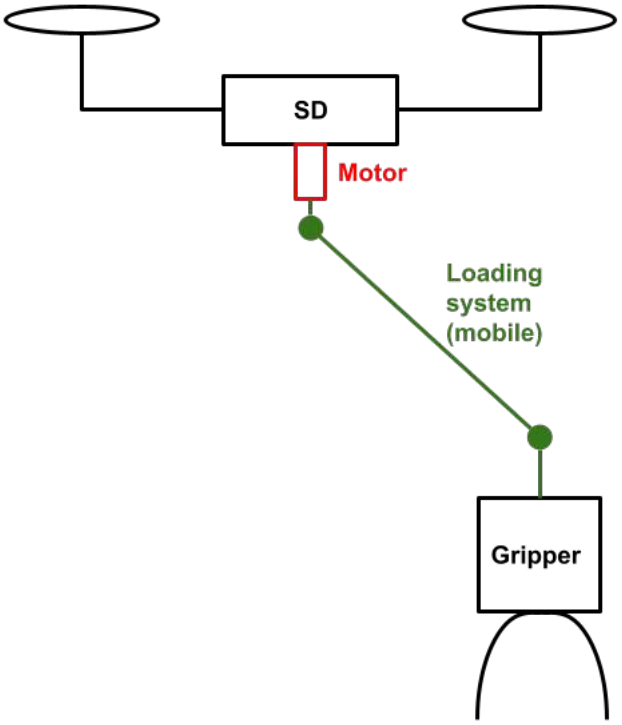


Figure 8: Deported loading system with complex geometry.

As previously mentioned, the primary challenge lies in decoupling the loading system from the closing mechanism to prevent any adverse effects on its speed. Existing literature highlights various approaches to address this issue, including the implementation of a dog collar mechanism, as discussed in the paper "*A Passive Closing, Tendon Driven, Adaptive Robot Hand for Ultra-Fast, Aerial Grasping and Perching*"[7]. However, this particular solution introduces additional complexity by necessitating an additional actuator to engage or disengage the coupling. In light of this, a simpler and more streamlined approach is deemed more suitable for our application. To strike a balance between the reloading system's complexity and overall performance, a compromise has been reached. The decision has been made to integrate a lightweight motor directly into the gripper mechanism. While this choice slightly increases the associated inertia, it significantly reduces the overall complexity of the system. By incorporating the motor within the mechanism, the reloading process becomes more streamlined and straightforward. This compromise ensures a practical and efficient solution while minimizing the impact on the overall performance of the gripper mechanism.

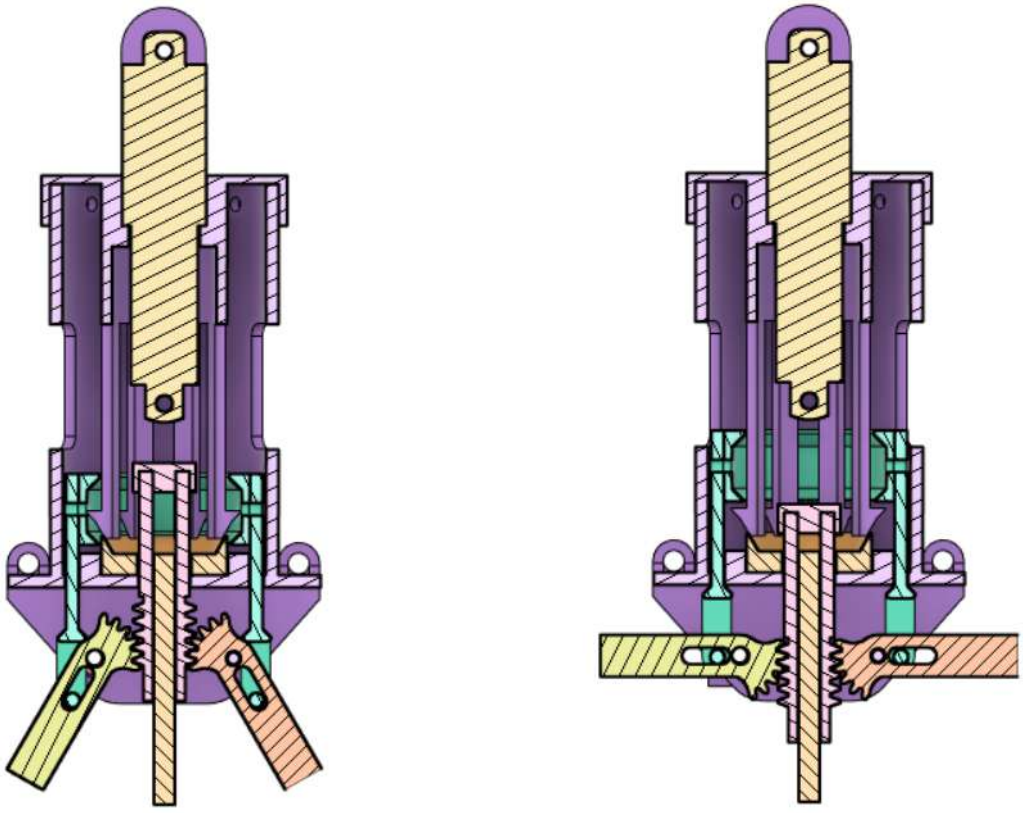


Figure 9: Open and closed states of the gripper with the loading system.

A clever design inspired by a wine corkscrew (bottle opener) has been developed for the gripper mechanism. The two fingers of the gripper are mounted on pinions and connected to a rack (pink part in **Figure 9**). When the fingers rotate, either to close or open, the rack moves in a vertical direction. This system exhibits reversibility, meaning that if the rack is moved up or down, the fingers will close or open accordingly. Leveraging this concept, a linear pusher (yellow part in **Figure 9**) can be incorporated inside the central cylindrical part of the device. This pusher is responsible for driving the rack downward, enabling the reloading of the mechanism. A notable advantage of this design is that once the device is loaded, the actuator can be retracted, effectively decoupling it from the rest of the mechanism during the closing operation. This feature enhances the overall functionality and efficiency of the gripper system.

- **Sizing**

  We are currently seeking to determine the appropriate sizing for both the spring and the motor. To aid in this process, we have a sketch available for reference (**Figure 10**). It should be noted that the problem exhibits symmetry, so for simplicity, we have depicted only one finger.

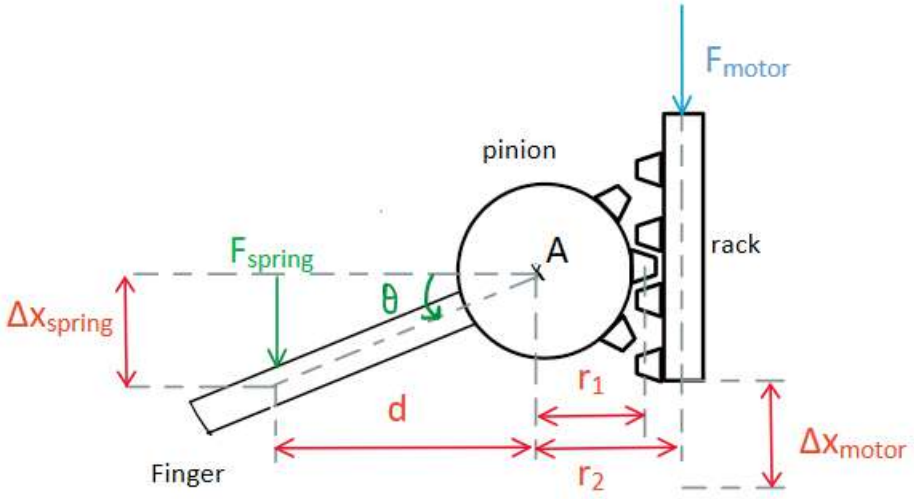


Figure 10: Sketch of the loading system.

By examining the system dynamics, we can establish a relationship between the force exerted by the spring and the necessary force from the motor. Specifically, we can calculate the torque at point A. For successful loading of the gripper, the sum of the torques must be negative, indicating that the torque generated by the motor surpasses the torque exerted by the spring. This condition ensures sufficient force to effectively reload the gripper mechanism, such as :

$$\Sigma M_A = -r_2 \times F_{motor} + d \times \cos(\theta) \times F_{spring} < 0 \tag{2}$$

This leads to the following result :

$$F_{motor} > F_{spring} \times \cos(\theta) \frac{d}{r_2} \tag{3}$$

The maximum force required occurs when $\theta = 0$:

$$F_{motor,max} > F_{spring} \frac{d}{r_2} \tag{4}$$

The force required to secure the plane hinges on the configuration of the gripper's finger (**Figure 11**).

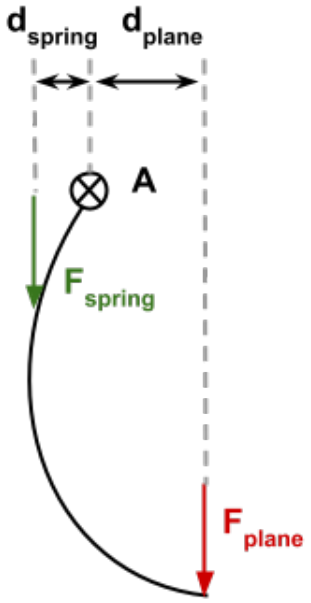


Figure 11: Sketch of the force applied on a finger.

We can sum up the moments at point A:

$$\Sigma M_a = d_{spring} \times F_{spring} - d_{plane} \times F_{plane} \tag{5}$$

If the contact points align vertically with the pivots ($d_{plane} = 0$), the plane's weight will not cause the claws to open. Additionally, ensuring effective adhesion between the gripper and the plane is crucial to maintain its position. These two aspects will be established through empirical testing, involving various finger shapes and materials.

In order to have control over the gripper's opening, it is essential to establish a relationship between the stroke of the motor and the corresponding angle of the fingers. The distance, denoted as d, required to achieve a rotation from an initial angle of $\theta_i = 0°$ to a final angle of θ_f can be calculated as follows:

$$d = \frac{\Delta x}{\tan(\theta_f)} \tag{6}$$

After considering various geometric constraints and searching for suitable spring components, we have identified a component[18] in (**Table 7**) that meets our requirements.

	D_{ext} **(mm)**	D_{int} **(mm)**	L_0 **(mm)**	L_{min} **(mm)**	F_{max} **(N)**	**k (N/mm)**
Spring	37.08	31.4	76.2	41.07	97.19	2.77

Table 7: Spring characteristics.

The selected spring has an operating range between 15mm and 5mm, providing a stroke of $\Delta x = 10mm$ and a maximum force of $F_{max} = 15 \times 2.77 = 41.55N$. For an opening of $\theta_f = 60°$, we can use **Equation** (6) to obtain $d = 5.77mm$. Using **Equation** (4), and fixing $r_2 = 16.5mm$, we then have:

$$\begin{aligned} F_{motor,max} > F_{spring}\frac{d}{r_2} &= 41.55 \times \frac{5.77}{16.5} \\ &= 14.53N \end{aligned} \tag{7}$$

Considering the potential presence of frictions and non-rigid parts in the system, it is prudent to choose a larger motor to account for these factors. This approach will provide an additional margin of safety and performance, enhancing the overall functionality of the system.
We can also compute the stroke of the motor Δx_{motor} which is:

$$\Delta x_{motor} = r_1 \times \theta_f \tag{8}$$

By fixing $r_1 = 10mm$, we obtain a stroke of:

$$\Delta x_{motor} = 10.47mm \tag{9}$$

After considering factors such as price and ensuring an adequate force output, we have identified a suitable linear actuator for our needs (**Table 8**). The chosen linear actuator (FIT0804[19]) possesses the necessary characteristics to fulfill the requirements of our gripper mechanism effectively.

	Voltage (V)	**Force (N)**	**Stroke (mm)**	**Speed (mm/s)**	**Mass (g)**
Motor	6	128	30	7	38

Table 8: Linear motor characteristics.

To provide feedback on the opening of the gripper since the motor lacks a linear encoder, we have incorporated a potentiometer onto the axis of rotation of one of the pinions. To meet the waterproof requirement, the potentiometer can be relocated to the top of the mechanism using parallel beams (**Figure 12**), as commonly seen in model making.

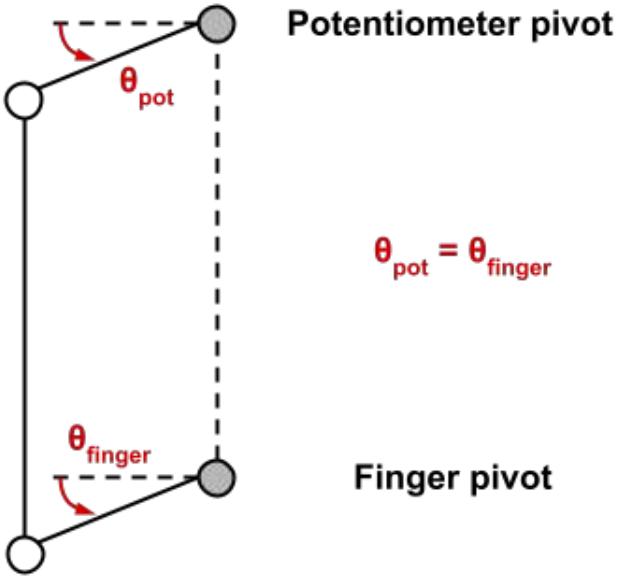
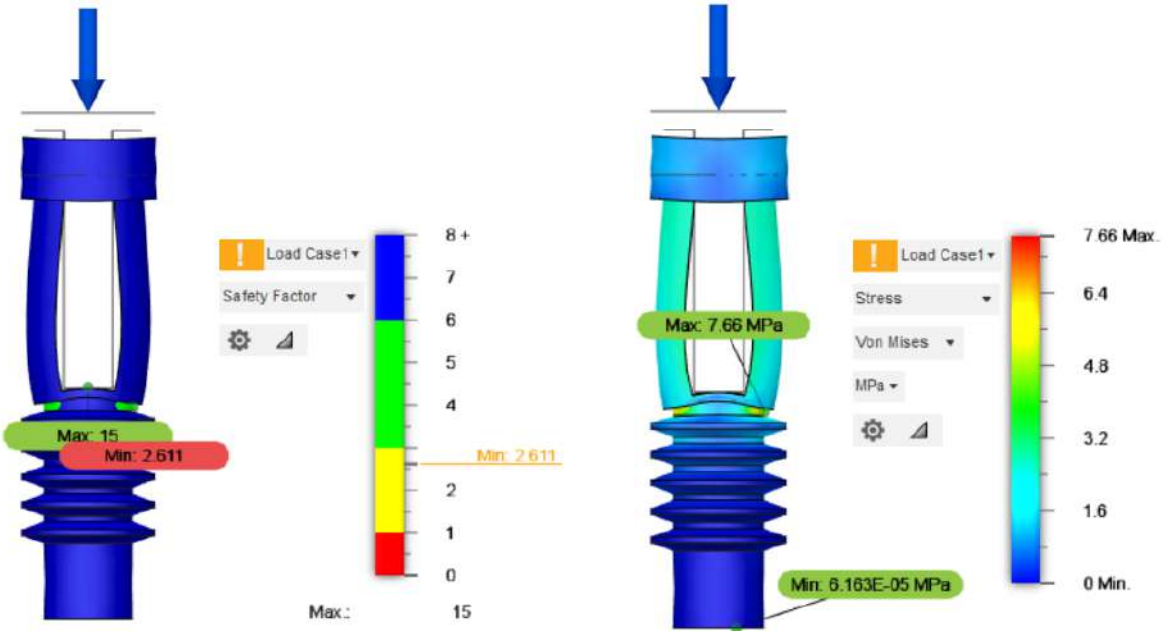


Figure 12: Parallel beams to deport the potentiometer and ensure waterproofing.

To ensure the structural integrity of the components under the high forces exerted by the motor, we conducted stress analysis on the parts with the highest constraints, i.e., the loading mechanism parts. This analysis aimed to determine the suitable material choice for optimal strength and durability. Ideally, the most cost-effective and time-efficient approach would be to 3D-print all parts. The material widely available for printing in our lab is mainly PLA. Although our simulation software lacks PLA in its material library, we can opt for ABS, which has properties quite similar to PLA. During the tests, we considered the worst-case scenario, where the motor applies its maximum force (128N) while the loading mechanism is already at the end of its stroke. The teeth of the mechanism remain fixed, and the force is applied on top of them. Additionally, the test was conducted using aluminum material. The results of the analysis are displayed in **Figure 13**.



(a) Safety factor. (b) Von Mises stress.

Figure 13: Stress analysis for the loading mechanism in ABS.

The safety factor is determined by dividing the material's yield strength by the Von Mises stress, which is a scalar value representing the combined tension and compression stresses at a particular point in the material, used in solid mechanics.

$$SF = \frac{\sigma_{yield}}{\sigma_{VM}} \tag{10}$$

In our software, the yield stress for ABS is set at 20MPa, resulting in a minimum safety factor of 2.6. When simulating with aluminum, the safety factor was found to be greater than 15 in all regions.
Despite the promising results, it is crucial to consider that the stress resistance of our 3D-printed parts in reality depends on the layer orientation, a factor not accounted for in the simulation, necessitating real-life testing for accurate assessment.

3.1.2 Robotic Arm

To successfully catch the TD, it is essential for the SD to be as close as possible. However, the challenge lies in the downwash effect caused by the propellers, which can disrupt the behavior of both drones. To address this issue, the gripper must be positioned at a certain distance from the propellers, but the exact distance is not well-known. According to the "*CFD simulation and experimental verification of the spatial and temporal distributions of the downwash airflow of a quad-rotor agricultural UAV in hover*"[20] paper, the airflow speed is proportional to the propeller diameter. In their case, with a rotor diameter of 0.6m, the airflow speed is approximately 2m/s at a distance of 1-1.5m from the drone (as seen in figure 5c). Closer to the propellers, at around 0.5m, the speed decreases, but fixing the mechanism at this distance would still mean encountering more disturbance approaching this zone. Since there is limited information available on the ideal distance, we need to conduct tests with different distances. For the calculations in this section, we will consider a distance of two times the diameter of the propeller, as it is sometimes accepted as a reasonable value.

To address this challenge, a solution is to design a robotic arm that positions the mechanism and the catch at a safe distance from the propellers. In this subsection, we will explore different robotic arm configurations. The key requirements that these arms must fulfill are as follows:

- The mechanism should maintain a minimum distance of at least 2 times the propellers' diameter when in the catching position, ensuring adequate clearance to prevent any interference between the mechanism and the propellers.
- The robotic arm should facilitate a safe landing of the SD, allowing for a controlled and stable descent to the ground.

To aid in the design process, a helpful sketch (**Figure 14**) depicting the various relevant dimensions can be utilized as a reference.

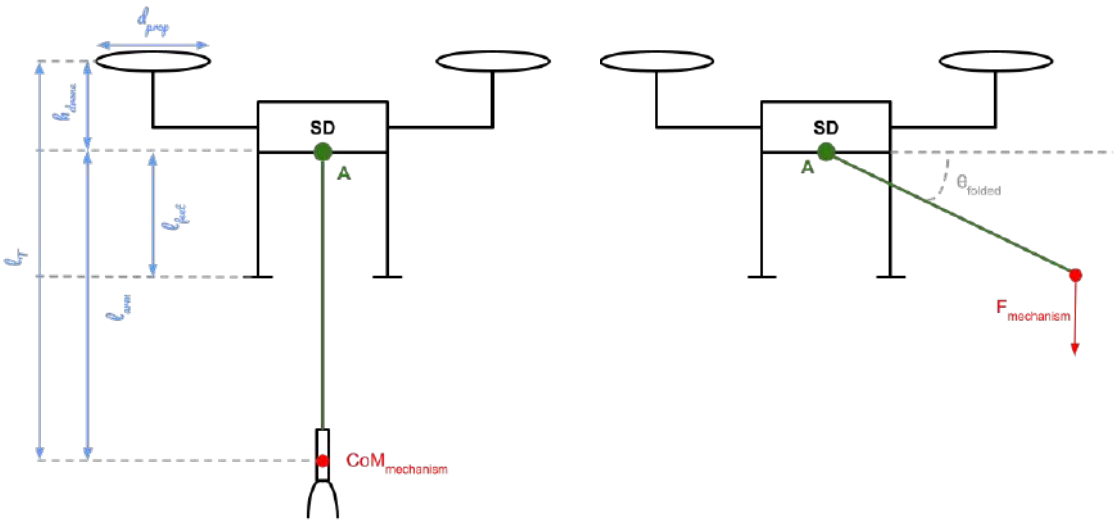


Figure 14: Sketches of the SD.

Name	Definition	Numerical value
d_{prop}	-	457.2 mm
l_T	$2 \times d_{prop}$	914.4 mm
h_{drone}	-	100 mm
l_{feet}	-	200 mm
l_{arm}	l_T-h_{drone}	814.4 mm
θ_{folded}	$\arcsin(\frac{l_{feet}}{l_{arm}})$	14.2 °

Table 9: Relevant dimensions of the SD.

The initial fundamental concept involves positioning a motor at point A to generate a torque for rotating the arm. We will assume that all the mass of the mechanism, arm, and TD is concentrated at the end of the arm, as depicted in **Figure 14**. By performing a quick summation of moments at point A and fixing $m_{total} = 1.2 + 0.4 = 1.6kg$, the following equation is derived:

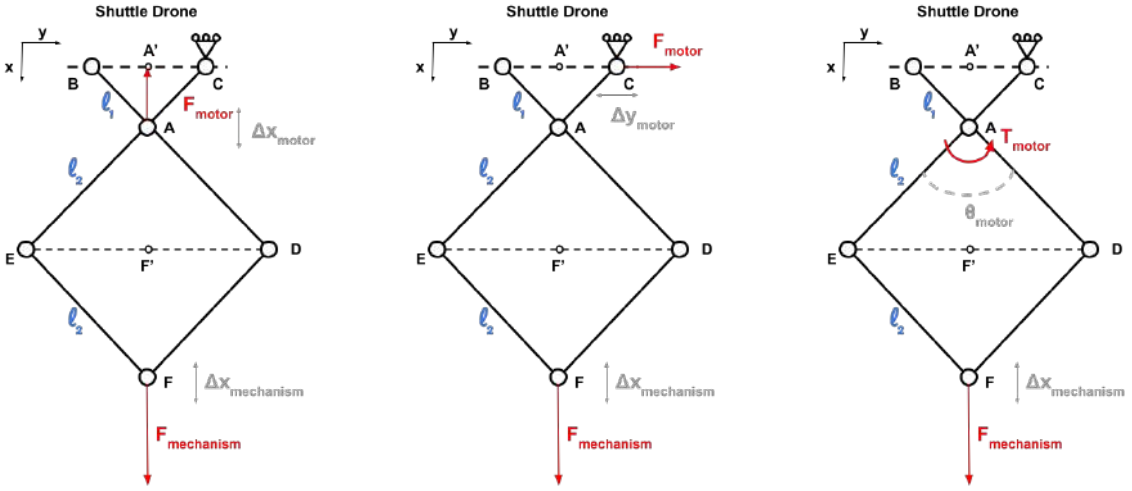
$$\begin{aligned} T_{motor} &> F_{mechanism} \times \cos(\theta_{folded}) \times l_{arm} \\ &> m_{total} \times g \times \cos(\theta_{folded}) \times l_{arm} \\ &> 12.4Nm \end{aligned} \tag{11}$$

The calculated torque value is unfeasibly high, making it impractical to find a motor capable of generating such torque without being excessively heavy. To address this issue, a different approach is needed. Instead of attempting to land the SD with the TD already gripped,

an alternative solution is to drop the TD in close proximity to the ground, reposition the SD, and then proceed with the landing. By adopting this revised strategy, the motor only needs to handle the mass of the mechanism itself, significantly reducing the load it needs to manage. Based on **Equation** (11), the required torque for this scenario is determined to be 3.1 Nm. However, this torque is still too high to fit the mass requirement. Consequently, it is crucial to explore and develop alternative solutions to ensure compliance with the desired criteria.

- **Scissor Lift**

  In the initial design approach, a scissor lift mechanism is employed to facilitate vertical movement of the mechanism. By adjusting the lengths of the small and long rods, a mechanical advantage can be achieved, resulting in a reduction of the force necessary to move the arm. Additionally, three distinct methods (**Figure 15**) of actuating the arm were thoroughly investigated and analyzed to identify the most suitable approach for the desired functionality.



(a) Vertical Force in A. (b) Horizontal force in C. (c) Torque in A.

Figure 15: Three ways of actuating the Scissor lift.

a) Vertical force

Using Thales' theorem, we can establish the relationship between the vertical movement of point A and point F. As the triangle ABA' and ADF' are rectangle, the ratio between the lengths is:

$$\frac{l_1}{l_2} = \frac{||A'A||}{||AF'||} \tag{12}$$

This leads to:

$$\begin{aligned} ||A'A|| &= x_A \\ x_F &= x_A + 2||AF'|| \\ \Rightarrow ||AF'|| &= \frac{x_F - x_A}{2} \end{aligned} \tag{13}$$

Using **Equation** (13) inside **Equation** (12), we obtain:

$$x_F = x_A \left(1 + \frac{2 \times l_2}{l_1}\right) \tag{14}$$

We can say that $x_F = x_{mechanism}$ and $x_A = x_{motor,a}$:

$$x_{mechanism} = x_{motor,a} \left(1 + \frac{2 \times l_2}{l_1}\right) \tag{15}$$

At equilibrium, the force at point A should counterbalance the applied load. We can calculate the relationship between the virtual work as follows:

$$\begin{aligned} \partial x_{mechanism} \times F_{mechanism} &= \partial x_{motor,a} \times F_{motor,a} \\ \Rightarrow F_{motor,a} &= F_{mechanism} \frac{\partial x_{mechanism}}{\partial x_{motor,a}} \end{aligned} \tag{16}$$

From **Equation** (15), we obtain:

$$\begin{aligned} \frac{\partial x_{mechanism}}{\partial x_{motor,a}} &= \left(1 + \frac{2 \times l_2}{l_1}\right) \\ \Rightarrow F_{motor,a} &= F_{mechanism} \left(1 + \frac{2 \times l_2}{l_1}\right) \end{aligned} \tag{17}$$

b) Horizontal force

The x coordinate of the motor is:

$$\begin{aligned} x_{motor,a} &= \sqrt{l_1^2 - (\frac{y_{motor,b}}{2})^2} \\ \Rightarrow x_{mechanism} &= \sqrt{l_1^2 - (\frac{y_{motor,b}}{2})^2} \left(1 + \frac{2 \times l_2}{l_1}\right) \end{aligned} \tag{18}$$

As before, the horizontal force should compensate the vertical load.

$$\begin{aligned} \partial x_{mechanism} \times F_{mechanism} &= \partial y_{motor,b} \times F_{motor,b} \\ \Rightarrow F_{motor,b} &= F_{mechanism} \frac{\partial x_{mechanism}}{\partial y_{motor,b}} \end{aligned} \tag{19}$$

$$\begin{aligned} \frac{\partial x_{mechanism}}{\partial y_{motor,b}} &= \frac{y_{motor,b}}{4\sqrt{l_1^2 - (\frac{y_{motor,b}}{2})^2}} \left(1 + \frac{2 \times l_2}{l_1}\right) \\ \Rightarrow F_{motor,a} &= \frac{y_{motor,b}}{4\sqrt{l_1^2 - (\frac{y_{motor,b}}{2})^2}} \left(1 + \frac{2 \times l_2}{l_1}\right) \end{aligned} \tag{20}$$

c) Torque

The x coordinate of the motor is:

$$\begin{aligned} \partial x_{mechanism} &= l_1 \times \cos\left(\frac{\theta_A}{2}\right) + 2 \times l_2 \times \cos\left(\frac{\theta_A}{2}\right) \\ &= \cos\left(\frac{\theta_A}{2}\right)(l_1 + 2 \times l_2) \end{aligned} \tag{21}$$

The torque required can be found using the virtual work again:

$$\begin{aligned} \partial x_{mechanism} \times F_{mechanism} &= \partial \theta_A \times T_{motor} \\ \Rightarrow T_{motor} &= F_{mechanism} \frac{\partial x_{mechanism}}{\partial \theta_A} \end{aligned} \tag{22}$$

With :

$$\frac{\partial x_{mechanism}}{\partial \theta_A} = -\frac{1}{2} sin\left(\frac{\theta_A}{2}\right)(l_1 + 2 \times l_2) \tag{23}$$

This leads to the following equation:

$$\Rightarrow T_{motor} = -\frac{1}{2} F_{mechanism} \times sin\left(\frac{\theta_A}{2}\right)(l_1 + 2 \times l_2) \tag{24}$$

If we select identical dimensions for all three types of configurations, the resulting outcomes are shown in **Table 10**.

	Vertical force	**Horizontal force**	**Torque**
Force/Torque (N or Nm)	33 N	30.4 N	1.5 Nm
Stroke (mm or °)	47.6 mm	115 mm	88°

Table 10: Summary of the performances of scissor lift designs.

Upon analyzing the results, it becomes evident that the horizontal force solution necessitates a prohibitively large stroke, rendering it unfeasible for practical implementation. On the other hand, the vertical force solution appears to be reasonable in terms of performance requirements. However, a significant drawback arises from the fact that point A undergoes both vertical and horizontal movements. This necessitates mounting the linear actuator on a rail, leading to a substantial increase in the complexity of the mechanism. Moreover, the scissor lift mechanism itself presents challenges. Point C must also be mounted on a rail, introducing multiple pivots in the arm that amplify both friction and vulnerability. In earlier equations, friction was neglected, but in reality, it cannot be disregarded as it significantly impacts the system's behavior. The presence of rails and multiple pivots further amplifies this friction, leading to a reduction in the mechanism's efficiency. Additionally, this configuration may suffer from instability. Consequently, the decision has been made to explore alternative solutions for the robotic arm, focusing on mitigating these complexities and addressing the stability concerns.

- **Linear Puller**

Another solution involves retaining the original concept of a single rod for the arm but implementing an actuation mechanism using a linear motor with a lever, instead of a rotating motor. This approach provides the flexibility to adjust the motor's performance by varying the distances at which the force is applied and where the motor is attached. By manipulating these distances, we can effectively tune the system to meet the desired performance requirements. **Figure 16** provides a visual representation of how this system operates.

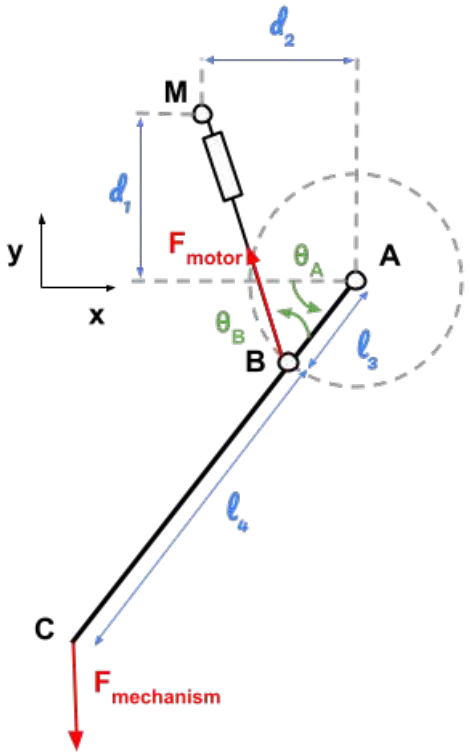


Figure 16: Sketch of the robotic arm using a linear actuator.

We want to find the force F_{motor} required to lift the arm to a desired angle $\theta_{A,f}$. We can start by a simple sum of torque at point A:

$$\begin{aligned} \Sigma M_A &= (l_3 + l_4) \times F_{mechanism} \times \cos(\theta_A) - l_3 \times F_{motor} \times \sin(\theta_B) < 0 \\ &\Rightarrow F_{motor} > \frac{l_3 + l_4}{l_3} F_{mechanism} \frac{\cos(\theta_A)}{\sin(\theta_B)} \end{aligned} \tag{25}$$

We can express the angle θ_B using the scalar product between vectors $\overrightarrow{BA}$ and $\overrightarrow{BM}$:

$$\theta_B = \arccos\left(\frac{\overrightarrow{BM} \cdot \overrightarrow{BA}}{||\overrightarrow{BM}||||\overrightarrow{BA}||}\right) \tag{26}$$

Considering the origin of our coordinates at point A, we obtain:

$$\begin{aligned}\overrightarrow{BM} &= \begin{pmatrix} d_2 + l_3 \times \cos(\theta_A) \\ d_1 + l_3 \times \sin(\theta_A) \end{pmatrix} \\ \overrightarrow{BA} &= \begin{pmatrix} l_3 \times \cos(\theta_A) \\ l_3 \times \sin(\theta_A) \end{pmatrix}\end{aligned} \tag{27}$$

The final angle can also be computed:

$$\theta_{A,f} = arcsin\left(\frac{l_{feet} - max(d_1, 0)}{l_3 + l_4}\right) \tag{28}$$

To determine the optimal set of parameters that minimizes the output force, an optimization process was employed. Initially, we manually selected certain parameters to obtain an approximate order of magnitude for the force. Subsequently, a motor capable of generating this force was chosen, and the optimization procedure was conducted while considering the geometrical constraints imposed by the motor, such as stroke length and dimensions.
Considering that the initially determined force was found to be less than 100N, we selected the same type of motor as the loading system but with a stroke length of 50mm. The stroke is computed as follows:

$$\begin{aligned} stroke &= ||\overrightarrow{MB}_i|| - ||\overrightarrow{MB}_f|| \\ &= ||\overrightarrow{MB}_{\theta_A=90°}|| - ||\overrightarrow{MB}_{\theta_A=\theta_{A,f}}|| \end{aligned} \tag{29}$$

The objective function used in the optimization process is as follows:

$$f(x) = \frac{l_3 + l_4}{l_3} F_{mechanism} \frac{\cos(\theta_A)}{\sin(\theta_B)} \quad = \frac{l_3 + l_4}{l_3} m_{mechanism} \times g \frac{\cos(\theta_A)}{\sin(\theta_B)} \tag{30}$$

The constraints of the system are as follows (the values are in meters):

1. $l_3 + l_4 + max(d_1, 0) = l_T - h_{drone}$
2. $\theta_B = \arccos\left(\frac{\overrightarrow{BM} \cdot \overrightarrow{BA}}{||\overrightarrow{BM}||||\overrightarrow{BA}||}\right)$
3. $\theta_A = \theta_{A,f}$
4. $stroke \leq 0.05$ (motor)
5. $||\overrightarrow{MB}_{\theta_A=\theta_{A,f}}|| = 0.1045$ (motor)
6. $0 < d_1 < 0.1045$ (motor)
7. $-0.15 < d_2 < -l_3$
8. $0.02 < l_3 < 0.2$

This optimization, performed with thee *minimize* function from the *Scipy.optimization* python library, leads to the results in **Table 11**.

d_1	d_2	l_3	Max Force	Max Stroke
0.0973	-0.0492	0.0492	56.1 N	50 mm

Table 11: Optimization results of the linear pusher robotic arm.

The torque and stroke of the motor are as in **Table 17**.

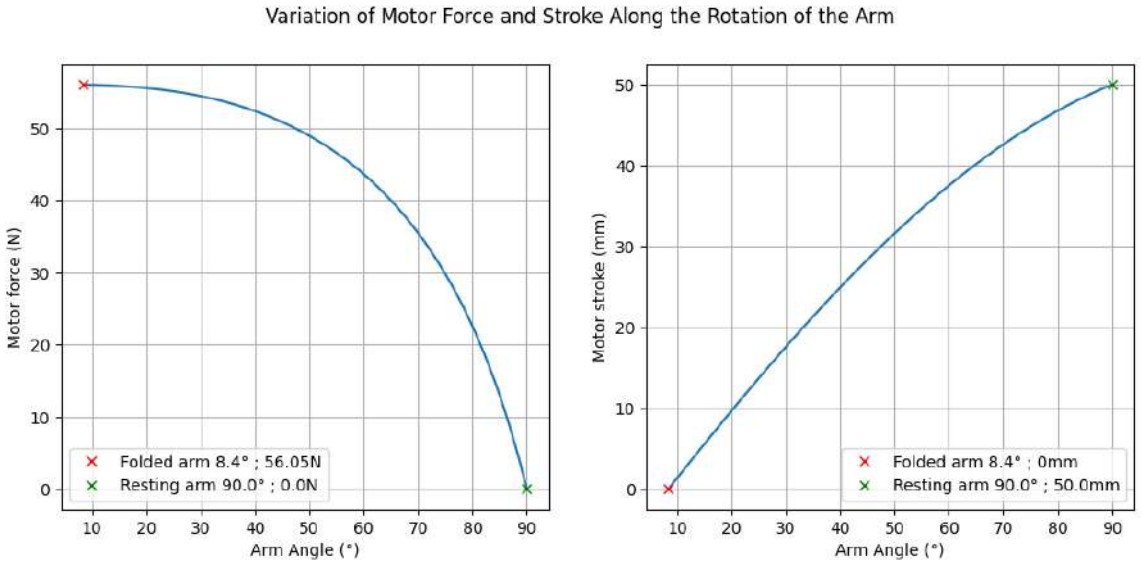


Figure 17: Variation of Motor Force and Stroke Along the Rotation of the Arm.

- **Toggle Joint**

  While the previous solution appears feasible in theory, we have decided to explore a third option to provide additional alternatives. This latest proposal involves the implementation of a toggle joint mechanism to reduce the torque required for arm lifting. **Figure 18** illustrates a sketch of the design, showcasing the configuration of this concept.

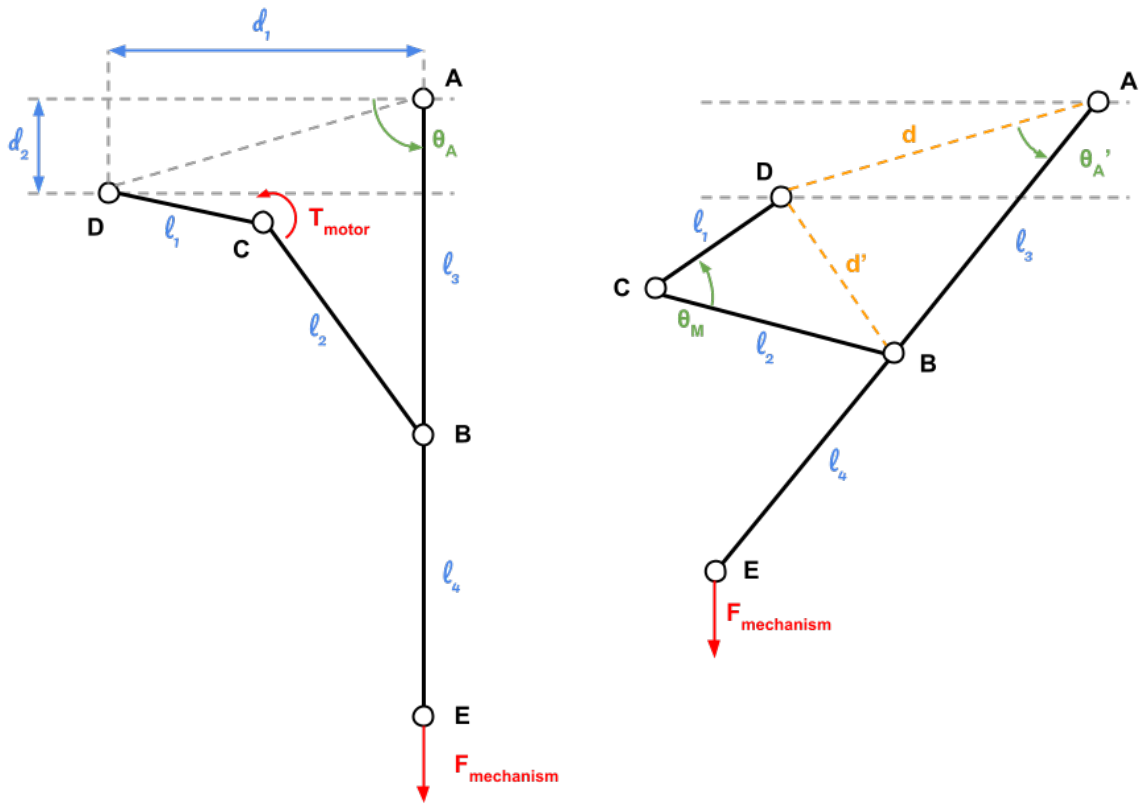


Figure 18: Sketches of the toggle joint design in resting (left) and folded (right) position.

To establish the relationship between the angle of the robotic arm, θ_A, and the motor angle, θ_M, we can utilize the laws of cosines in the quadrilateral ABCD. By applying the laws of cosines to this quadrilateral, we can derive the desired relationship.

$$\begin{aligned} d'^2 &= l_1^2 + l_2^2 - 2l_1 l_2 \cos(\theta_M) \\ &= l_3^2 + d^2 - 2l_3 d \cos(\theta'_A) \end{aligned} \tag{31}$$

This leads to the following equation:

$$\theta'_A = \arccos\left(-\frac{l_1^2 + l_2^2 - l_3^2 - d^2 - 2l_1 l_2 \cos(\theta_M)}{2dl_3}\right) \tag{32}$$

Finally, the arm angle is:

$$\theta_A = \theta'_A + atan\left(\frac{d_2}{d_1}\right) \tag{33}$$

The next step involves determining the torque required to lift the arm to a final angle, denoted as θ_f. By analyzing the mechanical forces and dynamics involved in the system, we can calculate the torque necessary to achieve this desired angle. Similar to previous calculations, we can employ the concept of virtual work to analyze the contribution of both the motor and the weight of the mechanism in determining the required torque.

$$\begin{aligned} T_{mechanism} \times \partial\theta_A &= T_{motor} \times \partial\theta_M \\ \Rightarrow T_{motor} &= T_{mechanism}\frac{\partial\theta_A}{\partial\theta_M} \\ &= (l_3 + l_4) \times \cos(\theta_A) \times F_{mechanism}\frac{\partial\theta_A}{\partial\theta_M} \end{aligned} \tag{34}$$

Due to the nonlinearity and complexity of the problem, running an optimization algorithm to find the optimal set of parameters becomes challenging. To address this issue, a grid search approach can be employed to compute the maximum torque obtained using different parameter combinations. The grid search method used to explore the parameter space takes into account several constraints.

- The first constraint is the requirement for a "knee" joint when the arm is in a vertical position. This ensures the stability and proper alignment of the arm in this specific configuration (**Constraint 1**).
- The second constraint involves the system's ability to reach the final angle, θ_f. It is crucial to ensure that the mechanism can achieve the desired angular position accurately and reliably (**Constraint 2**).
- The third constraint relates to the vector DC, which should not extend beyond the horizontal at the end of the arm's movement. This constraint guarantees that the arm remains within a safe range of motion (**Constraint 3**).

Based on the grid search analysis, the results from **Table 12** have been obtained:

$\mathbf{d_1}$	$\mathbf{d_2}$	$\mathbf{l_1}$	$\mathbf{l_2}$	$\mathbf{l_3}$	$\mathbf{l_4}$	**Max Torque**	**Max Stroke**
0.198	0.013	0.027	0.176	0.050	0.750	1.68 Nm	199°

Table 12: Grid search results of the toggle joint robotic arm.

In this option, the addition of a damping system to absorb the impact with the TD can be easily incorporated. This system is designed to mitigate vertical shocks, which can be compensated for by both drones. Although we assumed similar velocities, the

presence of a horizontal impact is still possible in real-world scenarios. To address this, two separate damping systems can be mounted on the arm. **Figure 19** illustrates the configuration, with the first damping system absorbing shocks from right to left. In this case, the toggle joint will move upwards but will be halted by the damping system. The second damping system will handle shocks from left to right, exerting a force directly on the vertical arm. This setup ensures that the arm remains vertical during the capture maneuver while effectively absorbing horizontal shocks in both directions. Both damping systems are not fixed to the arms but simply in contact with them.

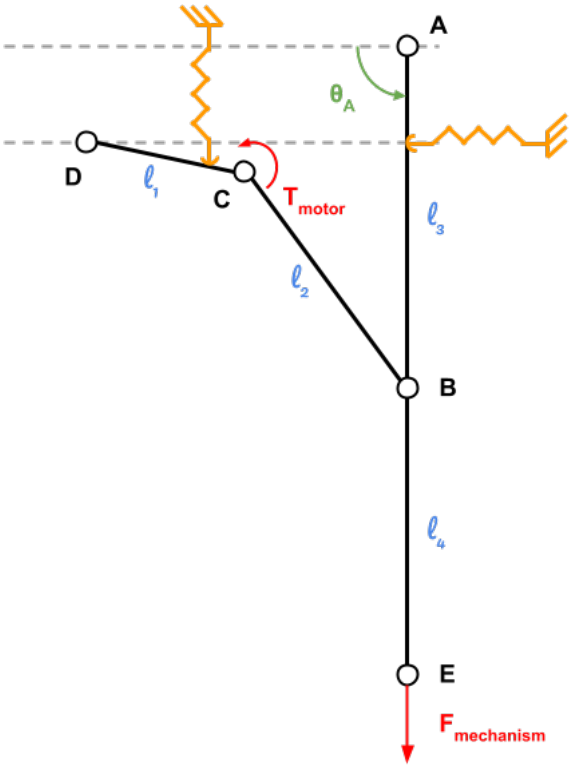


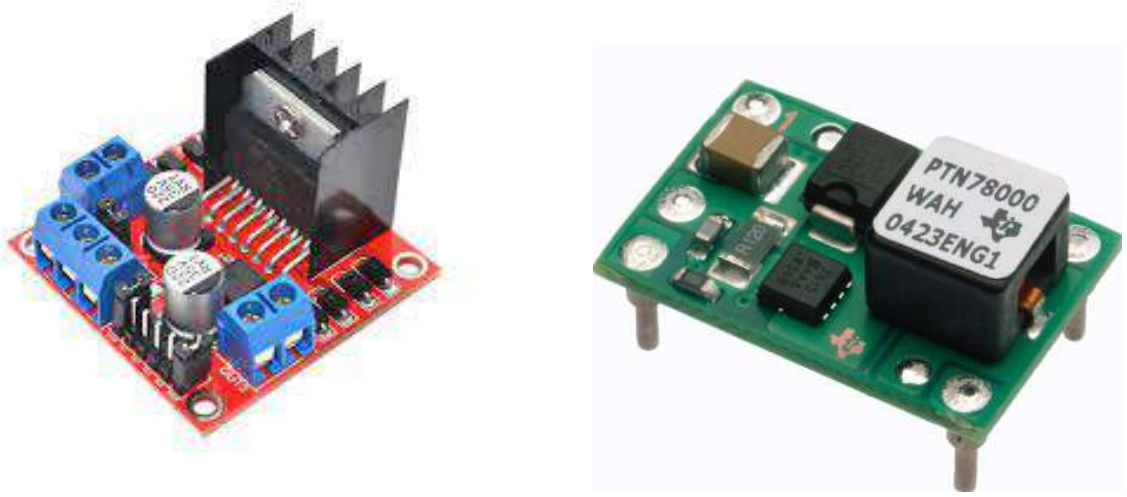
Figure 19: Sketch of the damping systems.

Given the requirement for high torque rather than high speed, selecting a motor with a substantial gear ratio is essential. Additionally, it is important to ensure a certain level of irreversibility in the motor, allowing the arm to remain in a resting position without the need for continuous power supply. However, if the motor is not reversible enough, the damping system will not be useful as the system will have no flexibility. The motor needs also to have torque big enough to compress the right damping system at the beginning of the arm movement as it is rotating first on the right before moving to the left.

3.1.3 Electronics

- **Hardware**

Within our mechanism, we manage two linear DC-motors, one for reloading the mechanism and the other for positioning the robotic arm. To control these DC motors effectively, we employ a motor driver. Our chosen motor driver is the L298N module, renowned for its capability to regulate both the speed and direction of two DC motors, making it an ideal choice for our application. The selected motor driver is a dual H-Bridge module, which facilitates the simultaneous control of two motors with voltage ranges between 5 and 35V, handling a peak current of 2A. Given that our motors operate at 6V and are expected to consume approximately 1A based on the datasheet, this motor driver is an ideal match. To control the direction of motor A, inputs 1 and 2 are utilized, while inputs 3 and 4 control motor B by managing the state of the H-Bridge switches. To modulate the motors' speed, ENABLE A and ENABLE B pins can be connected to a PWM signal, but as we prioritize position control over speed, we will not employ them. Instead, we will set jumpers on these pins to enable maximum speed. Based on the datasheet, the L298N module experiences a 2V voltage drop. Consequently, to effectively power the module, we need to provide it with an 8V voltage source.



(a) L298N Module. (b) PTN78000W Module.

Figure 20: Electronic modules used in our project.

Furthermore, in addition to the motor driver module, we have two potentiometers that serve as analog inputs to measure the position of the gripper and the arm. The control and processing of all these inputs and outputs are managed by an Arduino Nano board. The Arduino Nano can be powered by a voltage between 5V and 12V, enabling us

to utilize the same input voltage as the one used for the motor driver. To achieve a constant voltage source, we have a dedicated 12V power supply available on board. However, using a simple voltage divider with resistors won't suffice as the output voltage would be influenced by the current drawn by the load. To address this, we opted for a step-down switching regulator (PTN78000W), which ensures a stable and constant voltage output regardless of the load. The circuit diagram is shown in **Figure 21**.

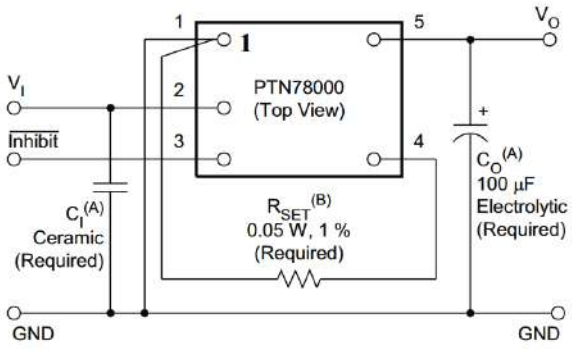


Figure 21: Ciircuit of the PTN78000W module.

The values of the input and output capacitors, as well as the resistor, are selected based on the recommendations and specifications provided in the datasheet of the PTN78000W step-down switching regulator[**noauthor_ptn78000w_nodate**]:

- Input ceramic capacitor

$$C_I = 2.2\mu F \tag{35}$$

- Output electrolytic capacitor

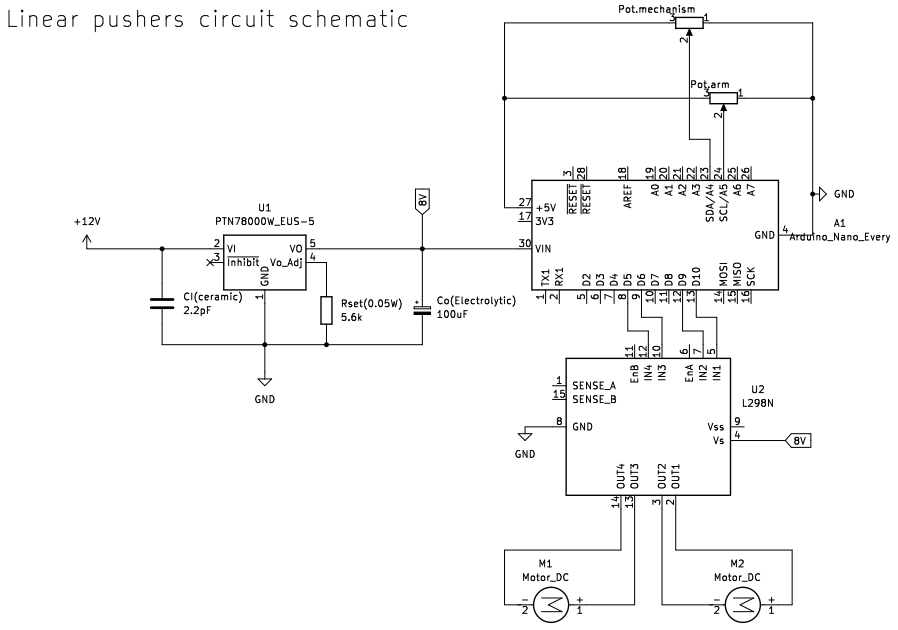
$$C_O = 100\mu F \tag{36}$$

- Setting resistor

$$R_{SET} = 54.9k\Omega \times \frac{1.25V}{V_O - V_{min}} - R_P \tag{37}$$

  where $V_{min} = 2.5V$ and $R_P = 6.49k\Omega$. This leads to $R_{SET} = 5.99k\Omega$. Due to availability reasons, we have opted for a $5.49k\Omega$ resistor, which results in an output voltage of approximately $8.23V$.

The complete circuit, including the step-down switching regulator (PTN78000W) and all other components, is displayed in **Figure 22**.



.9crop

Figure 22: Circuit schematics.

- **Software**

With the electronics hardware now in place, our focus shifts to controlling the two motors. The SD's on-board computer operates with ROS1, making communication with the Arduino Nano through ROS1 topics the most straightforward approach. For this purpose, there exists an existing package called "rosserial," which facilitates this communication. "Rosserial" functions as a protocol, encapsulating standard ROS serialized messages and multiplexing various topics and services over a character device, like a serial port.

The remaining ROS code is relatively simple. It involves publishing commands on a designated topic (*/arduino_cmd*) and subscribing to another topic (*/arduino_confirm*). This confirmation topic returns a signal once the required command has been successfully executed. To manage this process, we developed a node named *arduino_motor_cmd_node*. However, this node can potentially be replaced by a higher-level

node as this program integrates with the drone's overall program. To facilitate the launch of both the *arduino_motor_cmd_node* and the *rosserial_node*, a launch file is provided.

The Arduino implementation is straightforward, involving a simple program. This program establishes a subscriber and a publisher to handle incoming commands and transmit confirmations. Upon receiving a command, the program triggers one of FOUR functions: *deploy_arm_fct*, *fold_arm_fct*, *load_mech_fct*, or *state_mech_fct* contingent on the nature of the command. The confirmation process guarantees that we avoid executing multiple commands concurrently.

a) Deploy Arm

To deploy the robotic arm, the motor needs to move forward until the arm is completely vertical. Since the motor lacks an encoder, a potentiometer is utilized to determine the arm's angle. The potentiometer's value, ranging from 0 to 1023, is then converted into an angle due to the linear relationship between them. Once the angle value reaches 80°, the motor continues to apply force to the arm for an additional second before coming to a halt. We have selected 80° as the threshold value to mitigate potential errors. This decision acknowledges the conversion's imperfections; hence, even if the arm is fully vertical, the potentiometer might read a value of 89°. Once the motor ceases operation, a confirmation is generated and published on the corresponding topic.

b) Fold Arm

The same procedure is followed to retract the arm, but in this case, the motor is driven in reverse. The threshold value now relies on the arm's final angle, denoted as $\theta_{A,f}$. To account for potential variations, a tolerance value is introduced: $\theta_{threshold} = \theta_{A,f} + 5°$. Similarly, upon completion of the movement, the program emits a confirmation message.

c) Load Mechanism

To load the mechanism, the procedure remains the same, but a different potentiometer is utilized as an input. In this scenario, the objective is to confirm that the mechanism has been properly loaded. To achieve this, the threshold angle is set to $-10°$. Since the mechanism should be loaded at $\theta_{mech} = 0°$, this configuration ensures the trigger is properly engaged. Once this is accomplished, the pusher is retracted for a defined duration. Given the absence of an encoder, this period needs to be extended to account for the possibility of missing some steps in the DC motor. Following this, the program checks whether the gripper is open, as there's a possibility that the trigger mechanism might not block the system in

the open state. If it remains open, the same function is repeated. If it's closed, a confirmation is sent to the SD.

d) Check Mechanism State

This function serves to detect if the TD is caught. At regular intervals during the catching phase, the on-board computer can query the Arduino about the gripper's state. If the potentiometer angle is less than 5 degrees, the gripper is considered open. If the angle falls between two specific values (45 and 55 degrees in our tests), the plane is considered caught. However, if the angle is different, it usually indicates that the gripper is closed without having successfully grabbed the TD. In such cases, the reloading mechanism function can be invoked.

3.2 Results and Analysis

In this section, we will now assess the actual performances of our mechatronic design to determine if they align with our specified requirements. We will examine the outcomes related to the gripper and the robotic arm.

3.2.1 Gripper

Multiple iterations of prototyping were necessary to determine the appropriate specifications for our gripper to fulfill the requirements. The design discussed in **Section 3.1.1** was constructed using 3D-printed components to ensure consistency. The ultimate prototype design is depicted in **Figure 23**.

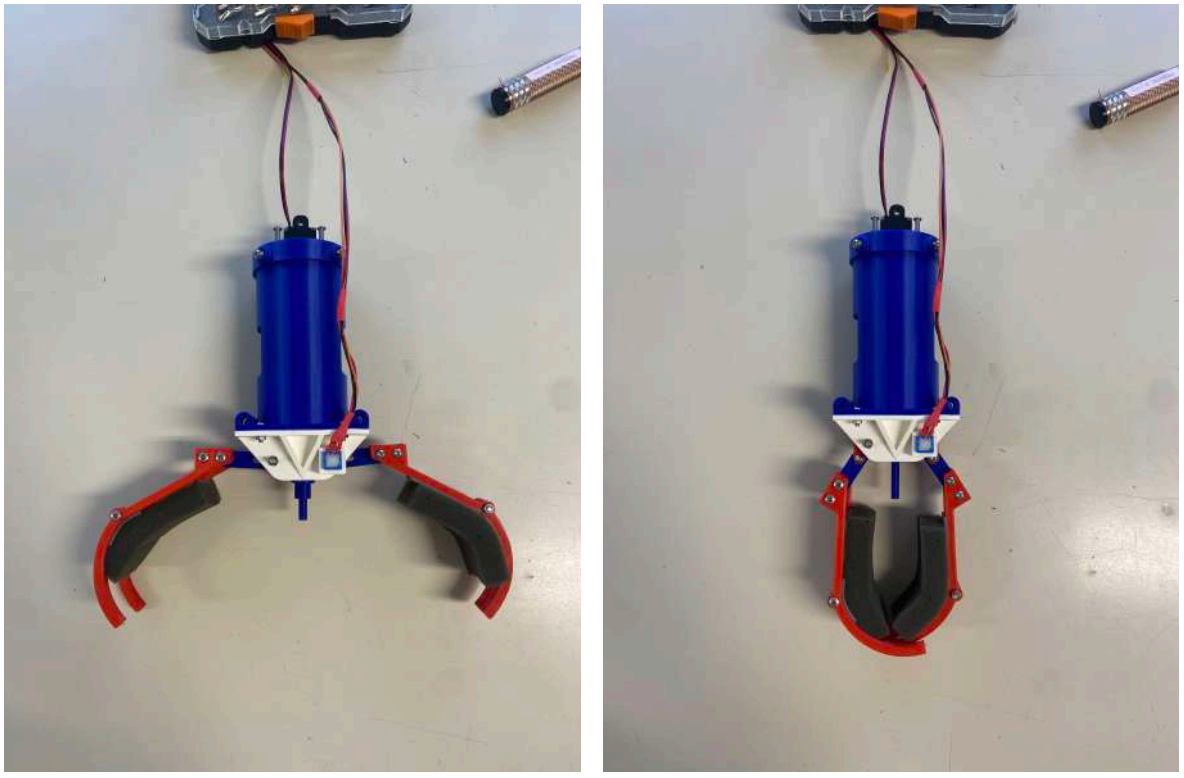


Figure 23: Prototype of the gripper.

Determining the force necessary to activate the mechanism is a critical aspect. As outlined in **Section 3.1.1**, the initial parameters for adjusting this sensitivity encompass the selection of materials and the cone angle of the central part. To retain production simplicity, we opted to continue using 3D-printed PLA components, refining them through light sanding to reduce friction. These materials and a 60-degree cone angle yielded promptly satisfactory outcomes. Building upon this foundation, we devised the trigger sensitivity adjustment system discussed in the same section. By collectively tightening or loosening four screws, we can fine-tune the

requisite trigger force.

To evaluate this system's efficacy, we gauged the force via a basic scale. It's important to note that this scale was rudimentary and not sufficiently precise to wholly rely on the results.

	0 mm		**8 mm**	
	Mass [g]	**Force [N]**	**Mass [g]**	**Force [N]**
Mean	110.7	1.1	216.5	2.1
std	40.9	0.40	23.3	0.23

Table 13: Force required to trigger the gripper.

Table 13 presents the mass required on the scale to initiate the mechanism's closure, along with the corresponding equivalent force. It's evident that when the screws are fully loosened (0 mm), the trigger force exhibits a substantial standard deviation, reflecting considerable variation in the outcomes. While some of this variability could stem from the scale's limited precision, the stark differentiation between the loose (0 mm) and tight (8 mm) configurations is evident, offering a substantial range for fine-tuning the trigger sensitivity. While further tightening of the screws is feasible, there's a caveat: we cannot guarantee the motor's capacity to consistently reload the mechanism under such conditions.

This observation prompts us to assess the reloading efficiency of our mechanism. The process outlined in **Section 3.1.3** yields a 100% success rate in achieving reloading. The section of the mechanism subjected to the most stress during reloading (refer to **Figure 13**) also demonstrates resilience, confirming the suitability of PLA as a suitable material for this application. The reloading procedure takes approximately 15 seconds to complete the gripper's opening.

As noted in the requirements, it's important for the end tip of the entire mechanism to be water-resistant. We previously discussed the option of relocating the potentiometer, either to the top of the mechanism or entirely on board the SD. Due to time constraints, we are currently retaining the potentiometer in its current position directly on the axis of rotation. Additionally, the motor we selected (FIT0803) has an Ingress Protection (IP) rating of IP54, which ensures it's safeguarded against dust and water to a certain extent.

Regarding the closing time, determining it is more complex. We attempted to use the potentiometer and Arduino to measure it, but the results were nonsensical (often showing null closing time). Nevertheless, by amalgamating this measurement with a visual assessment using a camera, we were able to conclude that the closing time is less than 200 ms.

To maintain the TD in position after catching it, the gripper's fingers play a crucial role. As discussed earlier, selecting the appropriate shape and material is essential. Various types of fingers were tested. The angle of the finger's shape (**Figure 24**) is a critical parameter to consider, ensuring it fits well with the aircraft body's contour.

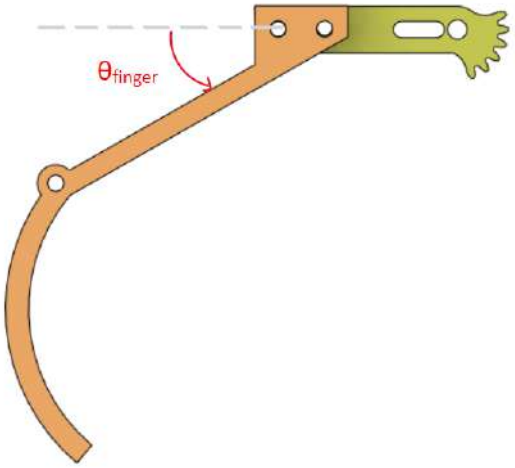


Figure 24: Gripper finger.

Since the final angle of the gripper is 60 degrees, having a small or zero θ_{finger} would result in inadequate grip on the TD's body. On the other hand, having an excessively large θ_{finger} could lead to a reduced space between the fingers when the mechanism is open, demanding higher accuracy to catch the plane. This challenge could be addressed by adjusting the mechanism's internal dimensions to achieve a larger angular stroke. Due to time constraints, we opted to retain the 60-degree closure angle.

Through a trade-off and selecting a finger angle of $\theta_{finger} = 30°$, along with adding a 2cm foam thickness, the gripper appears to perform well. The width between the two sides of the fingers when the gripper is open measures 20cm. Once the body of the TD is positioned between the two sets of fingers, the TD becomes trapped, and triggering the gripper becomes straightforward. A rubber part can also be added at the tip of the trigger to absorb eventual shocks and increase the contact area (**Figure 25**).

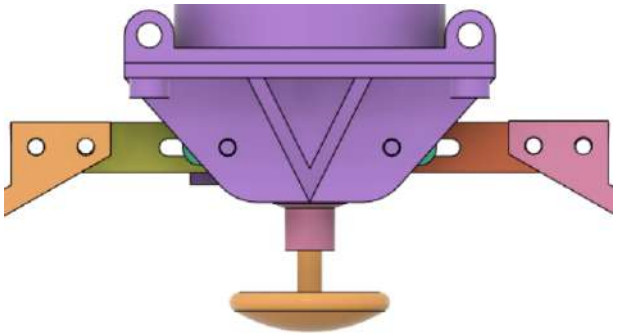


Figure 25: Rubber part on the trigger to absorb shocks.

The foam thickness depicted in **Figure 23** enhances the pressure exerted on the TD's body, thereby increasing friction. Consequently, the TD is held more securely, mitigating the risk of it slipping or falling off.

The significant advantage of these fingers is their easy adaptability, making it straightforward to replace them with different ones for other TDs.

The gripper's performance can be summarized in the table **Table 14**.

Total mass [g]	306
Force to trigger [N]	[1.1; 2.1]
Closing time [s]	<0.3
Reloading Success Rate	100 %
Reloading time [s]	15

Table 14: Gripper's performances

3.2.2 Robotic Arm

In **Section 3.1.2**, we presented various designs for the robotic arms and evaluated their performance. The scissor lift option was quickly eliminated due to its complexity in construction. The selection between the linear puller and toggle joint solutions will be influenced by the motor options available in the market. Let's briefly outline the performance requirements for each motor in both scenarios (**Table 15**).

	Linear puller	**Toggle joint**
Force/Torque	56.1 N	1.68 Nm
Stroke	50 mm	199 °

Table 15: Robotic arm motor requirements.

For the linear puller solution, an available motor with a 50mm stroke and a force generation capability of 128N was identified [21]. This motor's specifications seem to be adequate for our requirements. As for the toggle joint solution, an angular motor with a high torque was needed, prioritizing torque over rotational speed. Some servo motors with high gearbox ratios were found to be suitable due to their ability to hold positions without continuous power. However, a challenge arose as these motors provided static torque values in their datasheets, which might not align with their dynamic torque in practical application. In response, we procured two different motors: one with a static torque of 2.45 Nm [22] and another larger one with a static torque of 7.85 Nm. Both motors have a 270-degree stroke.

(a) Linear pusher with 128N force.

(b) Servo motor with 2.45Nm torque.

(c) Servo motor with 7.85Nm torque.

Figure 26: Motors considered to actuate the robotic arm.

To determine the optimal solution, we conducted testing with these motors. For the linear actuator, we fashioned a basic test setup where weights were hung at its end, attempting to move it across its entire stroke. The actuator demonstrated the ability to withstand forces exceeding 90 N, although we couldn't ascertain its exact limit due to the lack of additional weight. For the servo motors, we mounted them onto a horizontal plate, attached weights at the end, and attempted to achieve a 90-degree movement. In these tests, torque was calculated as the product of the force exerted by the weights and the distance from the center of rotation. The outcomes for the servo motors were less promising. The 2.45 Nm servo produced a dynamic torque of only 0.5-0.6 Nm, while the 7.85 Nm servo reached 1.5-1.7 Nm.

After comparing these results, the decision was made to proceed with the linear puller solution. This choice was based on the motor's greater safety margin, along with its lighter weight in comparison to the larger servo (220g vs 40g).

In **Section 3.1.2**, we conducted an analysis to identify the optimal parameters for enhancing motor performance. During these calculations, we determined the optimal arm angle using

Equation (28). However, we overlooked the dimensions of the gripper fingers in our initial considerations. To ensure an appropriate safety margin, we revised the final arm angle to be set at 0 degrees, resulting in a fully horizontal arm when folded. The updated dimensions are in **Table 16**

$\mathbf{d_1}$	$\mathbf{d_2}$	$\mathbf{l_3}$	$\mathbf{l_4}$	**Max Force**	**Max Stroke**
0.1045	-0.0437	0.0437	0.67	64.1 N	50 mm

Table 16: New geometric parameters of the linear puller robotic arm.

The arm's design is depicted as in **Figure 27**. We opted for a quadrilateral configuration with four rods to enhance robustness. The rods are made of carbon fiber to minimize the arm's weight and have a 5mm diameter. Additionally, we incorporated a contact point at the arm's end to prevent the TD from tilting forward, as elaborated in **Section 3.1.2**.

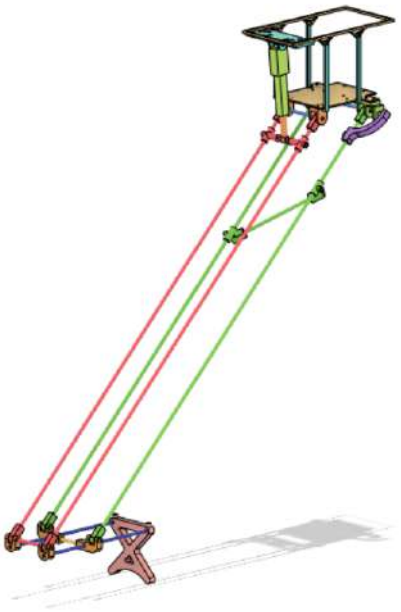


Figure 27: Design of the linear puller solution for the robotic arm.

The prototype was constructed and can be seen in **Figure 28**. The initial challenge we encountered was the arm's flexibility. Despite introducing cross sections between the rods, the system remained quite flexible. While this flexibility could potentially serve as an advantage by absorbing the impact during TD catching, it could also introduce accuracy issues during the catching phase. The airflow could cause the arm to bend, leading to a misalignment of the trigger tip from the expected position. If we observe this behavior during trials, addressing this concern might involve developing a way to locate the trigger tip, either through computer

vision or by integrating a sensor like an IMU within the mechanism.

If the arm's flexibility proves problematic during flights, increasing the diameter of the carbon rods might be necessary to enhance rigidity. However, this adjustment would also lead to an increase in the arm's weight, which currently stands at 280g without electronics.

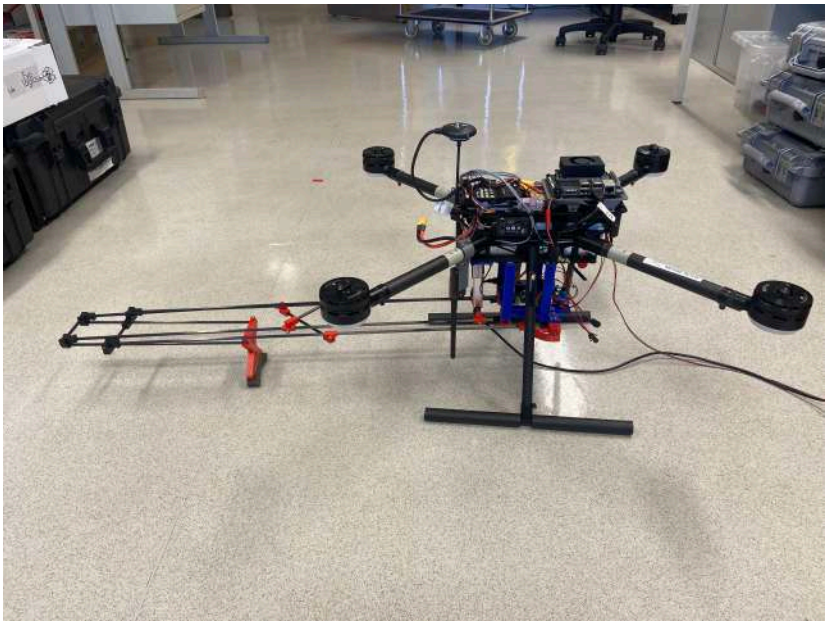


Figure 28: Robotic arm prototype.

Another potential challenge we could encounter is the sensitivity of the TD to wind conditions. Given its large wingspan and the combined weight of the SD and TD, maneuvering could prove difficult. The airflow might lead to unpredictable TD behavior, potentially causing damage to the robotic arm. To mitigate this risk, several solutions can be explored.

Firstly, as previously mentioned, increasing the diameter of the rods could result in a sturdier arm. Additionally, assessing whether the downwash perturbations from the SD are as significant as anticipated could offer insights. If not, shortening the arm's length could be considered. Lastly, the TD could be actively controlled even after being caught. This approach would involve the TD maintaining a consistent attitude, thereby working in tandem with the SD to facilitate a smoother landing process.

To fix the arm to the SD, we can adopt a similar design that's already present on the drone. The current design, depicted on the left side of **Figure 29**, comprises a carbon plate secured to the base of the SD using four aluminum components. However, it's necessary to adjust the dimensions of the carbon plate and the aluminum parts. Although we couldn't machine these

components due to time constraints, we temporarily 3D printed them. The 2D schematic of the aluminum parts is available in the appendix, and the CAD file for cutting the carbon plate will be provided in a separate GitHub repository. We conducted a stress analysis to assess whether these parts could withstand the weight of the plane, and the safety factor was consistently above 15 in all areas. Larger 3D printed parts were employed, which also demonstrated the capability to withstand this force.

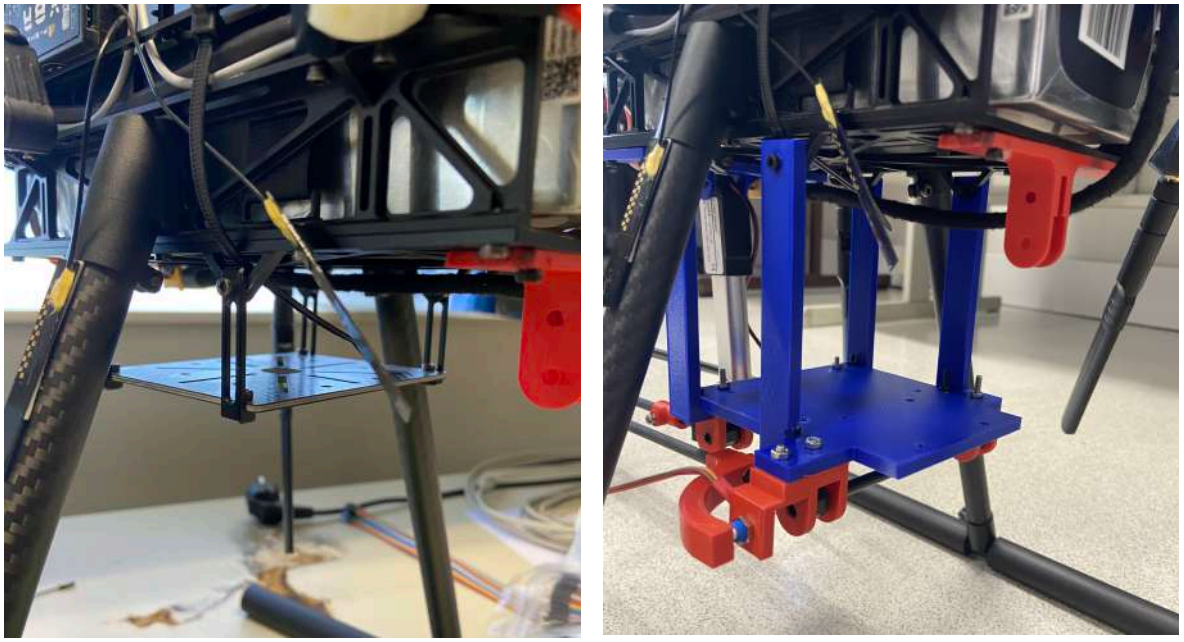


Figure 29: Parts to fix the robotic arm to the SD.

3.2.3 Electronics

The electronics results were swiftly assessed by soldering all components onto a universal PCB and measuring the output voltage of the step-down switching regulator. As anticipated, the output was accurately measured at 8.23V for an input voltage of 12V. During operation, each of the motors consumes less than 1 Amp. The total weight of the board is 89g.

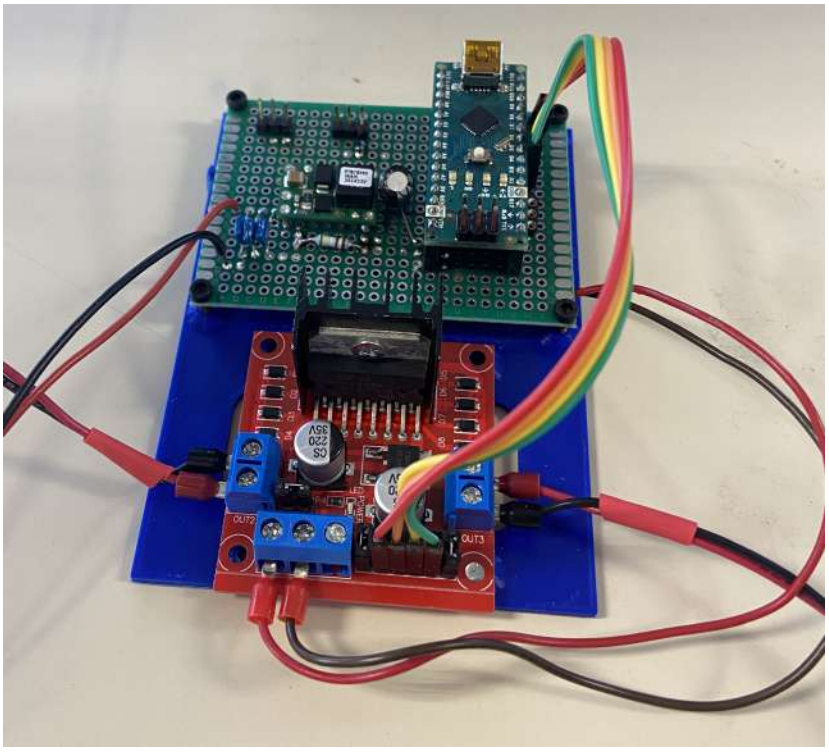


Figure 30: Electronic circuit.

Concerning the software aspect, all components are functioning according to the plan. The ROS1-ROS2 bridge effectively translates topics between different ROS versions, enabling motor control. Both the reloading algorithm and the arm folding/deploying algorithm are operational. The average time for arm deployment is 7 seconds, while the average time for arm folding is 9 seconds. Since the motors have a substantial gearbox ratio, they are non-reversible and only consume energy during movement, ensuring minimal power consumption for maintaining a desired position.

3.2.4 Target Drone Catching Performance

The ultimate outcome of our mechatronic design, which encompasses the gripper, the robotic arm fixed to the SD, and the integrated electronics, can be visualized in **Figure 31**.

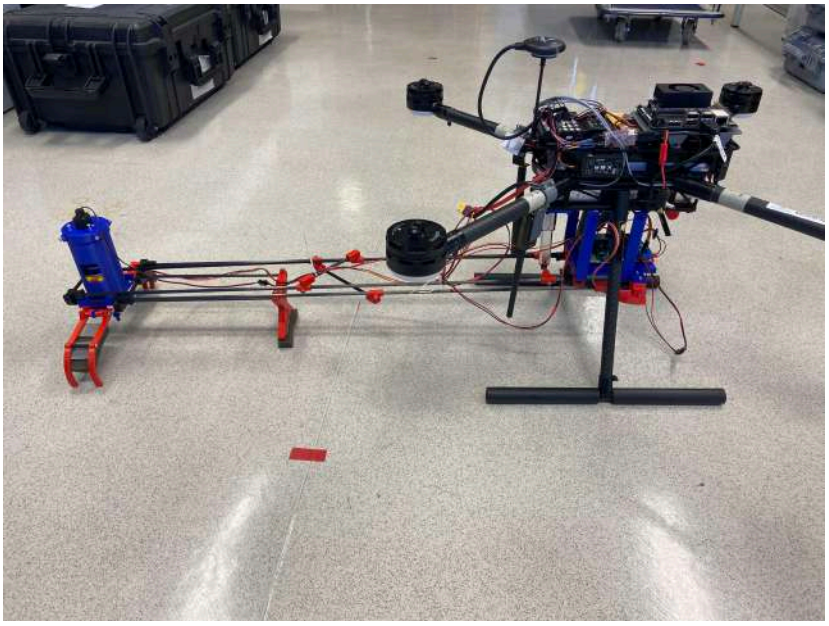


Figure 31: Full design of our mechanism.

Due to a scheduling conflict, actual flight tests couldn't be conducted during the duration of this thesis. These tests were unfeasible in the final month of the study due to the mechanism, robotic arm, and electronics not being ready in advance. Consequently, to assess the system's performance, we had to resort to manually catching the TD (**Figure 32**). In this process, the catching system is positioned beneath the SD and manually maneuvered near the TD. The TD itself is positioned slightly above the ground to facilitate the catching procedure.

Figure 32: Screenshot of a manual capture of the TD.

The objective of this test was to carry out the complete process, beginning with the SD taking off and deploying its arm. It then involved localizing the TD and activating the gripper. Subsequently, the SD would transport and release the TD near the ground by opening the gripper. Finally, it would retract its arm and initiate the landing. These tests were conducted

multiple times and consistently yielded successful results.

In the future, the following tests should be conducted:

- TD placed on tripods at a height of 1.5 meters above the ground, while the SD is controlled to attempt capture.
- TD positioned on tripods at a height of 1.5 meters above the ground, while the SD is autonomously controlled to attempt capture.
- TD in autonomous flight, while the SD is autonomously controlled to attempt capture.

4 Target Localization

To catch the TD, the SD must accurately locate it and position the tip of the mechanism directly above it. This section discusses the various methods employed to localize the target drone.

4.1 Methods

Each technique comes with its advantages and disadvantages, which depend on factors like the distance between the SD and the TD, as well as their relative velocities.

To aid us in this endeavor, we have access to several sensors. The primary one is an Intel® RealSense™ Depth camera, which can be either the d435i or the d455 model. These cameras employ RGB imaging and stereo techniques to gauge depth. They are also fitted with their own IMUs. Their performance characteristics are outlined in **Table 17**.

	D435i	**D455**
Use Environment	Indoor & Outdoor	Indoor & Outdoor
Depth FOV (HxV)	87° × 58°	87° × 58°
Depth Resolution	Up to 1280x720	Up to 1280x720
Depth Frame Rate	Up to 90 fps	Up to 90 fps
Depth Accuracy	<2% at 2m	<2% at 4m
Minimum Depth Distance at Max Resolution	∼28 cm	∼52 cm
Ideal Range	.3 to 3 m	.6 to 6 m
RGB Sensor Technology	Rolling Shutter	Global Shutter
RGB Resolution & Frame Rate	1920 × 1080 at 30 fps	1280 × 800 at 30 fps
RGB FOV (HxV)	69° × 42°	90° × 65°

Table 17: Intel® RealSense™ Depth cameras performances

As illustrated in **Table 17**, the D455 camera exhibits superior performance compared to the D435i. It boasts a larger Field of View (FOV), a greater operational range, and enhanced accuracy. While its resolution is slightly smaller, it remains sufficient for our intended application. Another notable advantage of the D455 is its RGB Sensor Technology. The use of a Global Shutter means that all pixels capture light simultaneously, as opposed to a Rolling Shutter where they capture light row by row (or column by column). This Global Shutter configuration is particularly advantageous for fast-moving objects, such as in our case.

To secure the camera onto the drone, we created an adapter featuring various mounting holes, allowing for adjustments to the camera angle. Given that the TD drone aligns with the roll

axis of the SD, we positioned the camera vertically to leverage its broader Field of View (FOV) along the horizontal axis (**Figure 33**).

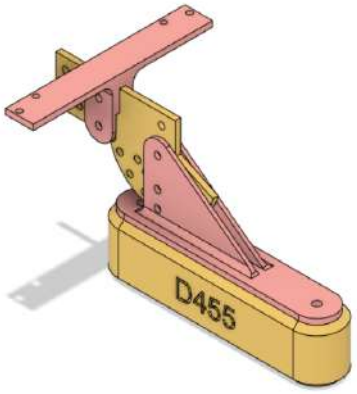


Figure 33: Assembly to attach the camera to the drone.

As previously mentioned, we will now delve into various methods for localizing the TD.

4.1.1 GPS

The first method we consider for long-range localization involves utilizing GPS technology. This approach is handled by a separate division within the team, and while we won't go into the specifics of its development here, it's important to acknowledge its existence. This method capitalizes on the Global Positioning System to determine the precise position of the TD in relation to the SD. In our case, the GPS-based method offers a precision of around 2.5 meters and 0.3 degrees[**noauthor_drone_nodate**]. However, since we aim to measure the relative position and orientation between the two drones, the accuracy is effectively doubled. It serves as an effective means of locating the TD when it's positioned outside the FOV of the SD, especially during long-range interactions. However, one of the drawbacks of this method is that its accuracy diminishes considerably when the TD comes within close proximity to the SD. As such, relying solely on GPS for catching the TD isn't feasible due to its limitations in accuracy at short distances.

4.1.2 ArUco Markers

For situations requiring closer range localization, an alternative technique comes into play. Leveraging the work of a former student at the DSOR Lab, Fransisco Azevedo[**azevedo_francisco_nodate** who contributed to the REPLACE project[14], offers valuable insights. Within this project, a package's position is determined using a combination of ArUco markers and a camera.

ArUco markers are a type of fiducial marker used in computer vision applications for pose estimation. These square markers are printed with a pattern of black and white squares that form a unique code. A camera captures an image containing these markers, and the code is decoded to determine the marker's identity and orientation in 3D space. ArUco markers are widely used for tasks like camera calibration, object tracking, and robotics. The distinct patterns and ease of detection make them an efficient tool for providing accurate position and orientation information. As the theory of ArUco markers has already been explained in Fransisco's master's thesis, we will not go more into detail about it here.

a) ArUco tracking

We opted to utilize ROS2 for detecting the TD using ArUco markers, even though the majority of the programs on board rely on ROS1. This decision was based on the availability of multiple existing packages in ROS2, specifically for the Intel® RealSense™ camera and ArUco tracking. To establish communication between ROS1 and ROS2, a ROS2 package called *ros1-bridge* was utilized. This package facilitates the conversion of ROS1 topics to ROS2 and vice versa. It offers the flexibility to convert either all topics or only those specified in the *ros1_bridge.yaml* configuration file.
The ArUco tracking program's architecture can be summarized with the following ROS graph (**Figure 34**), where ellipses represent nodes, rectangles symbolize topics, and arrows represent message flow.

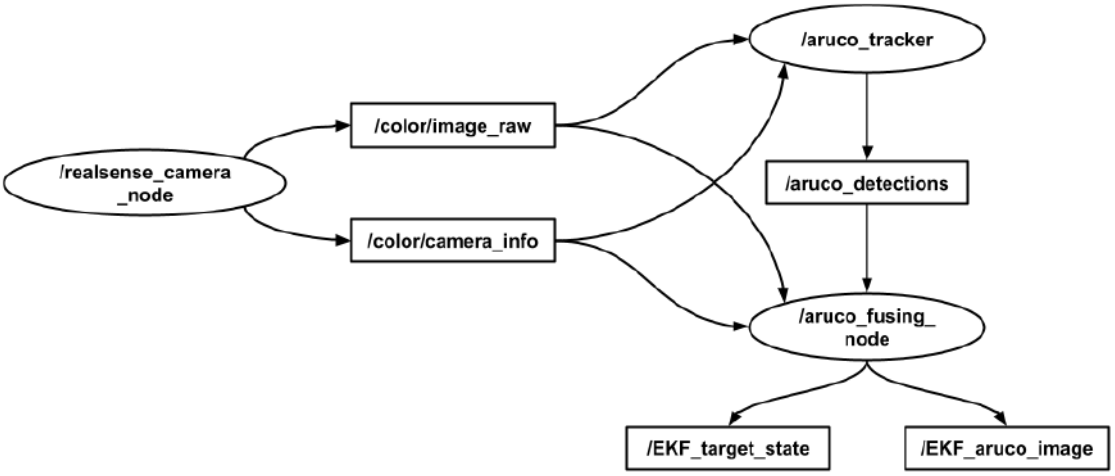


Figure 34: ROS graph of the ArUco detection programs architecture.

The script can be launched via the *realsense_aruco_detection.launch.py* file, which initiates three interrelated nodes.

- The */realsense_camera_node* acquires the image from the onboard Intel RealSense Depth Camera and publishes both the image and the camera parameters (intrinsic and distortion) on two topics: */color/image_raw* and */color/camera_info*. This node's code is provided by Intel on GitHub[23].
- The */aruco_tracker* node subscribes to the aforementioned topics and performs the detection of ArUco markers visible in the image. Originally authored by Baej Sowa[24] and modified by Marcelo Jacinto[25], a PhD student at IST, and myself, this node offers versatility as it can detect ArUco markers and boards for various projects. It loads board descriptions from a file (*board_descriptions.yaml*) and proceeds to detect these boards in the frame. A board is defined by parameters such as the number of markers on the x-axis, the number of markers on the y-axis, the ID of the first marker, the size of the markers, and the separation between them (**Figure 35**). Additionally, the node provides the option to plot the axis of the ArUco at the center of the board or with an offset in the x, y, and/or z directions (with the origin of the board at the top left corner of the first marker). In our project, we have configured the offsets of each board to ensure that the detected point coincides with the center of the TD. By aligning each board's detection at the center of the TD, the fusion of this data becomes more straightforward and allows for easier integration and analysis of the information.

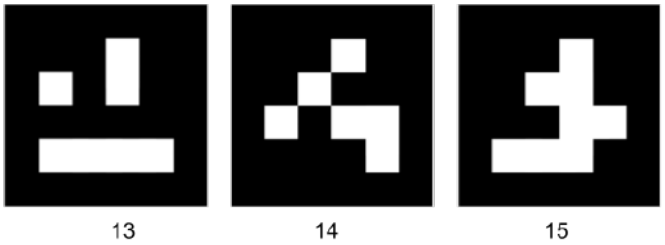


Figure 35: Example of ArUco board with 3 markers in the x direction and 1 in the y, starting with the ID 13.

Once the positions of all detected markers and boards are determined, they are published on the */aruco_detections* topic using the *ArucoDetection* message. This customized message gathers the pose (position and orientation) of all detected markers and boards along with their respective IDs/names.

- The */aruco_fusing_node* subscribes to the */aruco_detections* topic and performs the fusion of the detected board's poses to obtain the estimated position of the TD. The fusion process utilizes an Extended Kalman Filter (EKF) to refine the attitude of the TD. More detailed information on the EKF implementation will be provided later. Once the EKF has predicted the state of the TD, it is published on the */EKF_target_state* topic. Additionally, the node publishes the image with the axes of the TD overlaid on the */EKF_aruco_image* topic.

b) ArUco boards

In our project, we plan to implement multiple ArUco boards on the TD to enhance detection reliability. Having multiple boards will enable us to detect the target even if some markers are not visible within the frame. This approach provides robustness and improves the overall accuracy of the TD detection process.
To explore different detection configurations, we will experiment with various board sizes and numbers of markers for each board. This testing process will help us identify the most optimal setup that ensures consistent and accurate TD tracking under various conditions and scenarios.

Figure 36: Example of ArUco boards disposal on the target drone.

On **Figure 36**, each green rectangle represents a board, which can be composed of either a single ArUco marker or multiple ArUco markers arranged in a grid pattern.

Aruco markers provide an effective way of object tracking using a camera. Their precision is contingent on the camera's capabilities, which will be ascertained subsequently. It is certain that accuracy is influenced by object velocities and distances. The velocity concern can be mitigated by the D455 camera due to its Global Shutter technology, which enhances frame quality for fast-moving objects. Nevertheless, an alternative method for detecting the TD within mid-range needs to be identified.

4.1.3 Depth Detection

a) Depth Clustering

To detect the TD under circumstances where ArUco markers are poorly detected and GPS accuracy falls short, one potential approach is to utilize the depth measurement feature provided by the Intel® RealSense™ Depth Camera. The depth measurement produces an image with pixel values ranging from 0 (indicating no detection in that pixel) to the maximum detectable distance (approximately 6m, as specified in the datasheet). To determine the 3D position from a depth measurement, we must account for the misalignment between the depth image and the color image, as they use different reference frames. This alignment is achieved by synchronizing the depth image with the color image, and fortunately, the ros2 node responsible for launching the RealSense™ camera already publishes the aligned depth image on a topic. The next step involves converting a depth measurement within a specific pixel into a corresponding 3D point. To achieve this, we must perform a process called "deprojection," which involves converting a pixel's coordinates (u, v) into its corresponding 3D position. The projection from 3D to 2D is straightforward and involves transforming 3D coordinates (X, Y, Z) into 2D pixel coordinates (u, v) using a camera matrix and distortion coefficients.

$$\begin{aligned}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= K \cdot (R|T) \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\ &= \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}\end{aligned} \tag{38}$$

where K in the intrinsic parameters matrix, and $(R|T)$ is the extrinsic parameters matrix.On the other side, deprojection requires the inverse of this transformation, which is a more complex task and involves solving the equations that map 2D pixel coordinates back to their original 3D positions in the real world. Fortunately, the RealSense library already provides this transformation, so we don't need to concern ourselves with it. Now that we have the ability to convert depth measurements to 3D positions, we can use this information to detect the target object.
By examining the depth camera's field of view, we can devise a method to determine the pose of the TD.

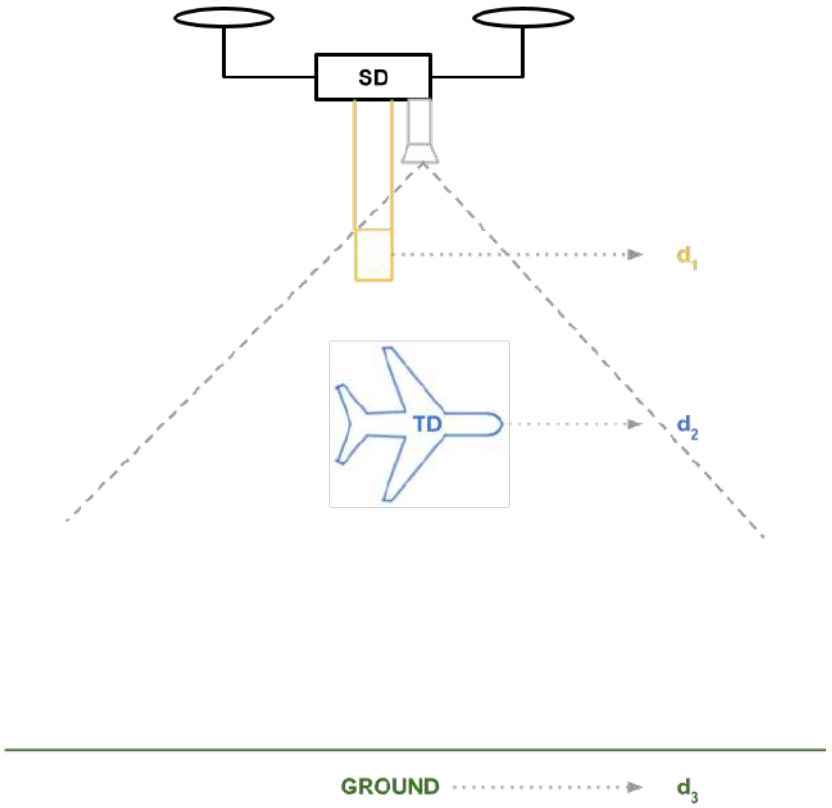


Figure 37: Sketch of the Field of View (FoV) of the depth camera.

Since the TD is expected to fly well above the ground, the camera's field of view should primarily capture three distinct elements: the mechanism (shown in yellow in **Figure 37**), the TD (in blue), and the ground (in green). These elements will be significantly separated from each other, resulting in a depth pixel density that resembles the representation in **Figure 38**

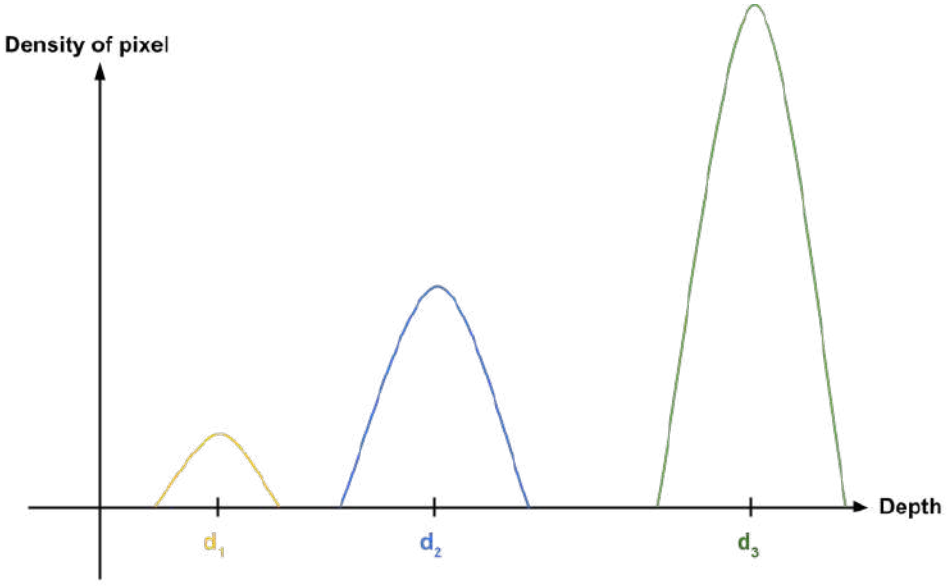


Figure 38: Depth pixel density graphic.

Our objective is to separate the blue point cloud, which represents the TD, from the rest of the scene. To achieve this, we can employ a technique called clustering, which falls under the category of unsupervised machine learning. Clustering involves grouping data points together based on their similarities. In this context, the main similarity appears to be the depth of each point. Various clustering algorithms exist, and for our purposes, we will concentrate on two of them: Gaussian Mixture Model (GMM) and K-means. **Table 18** presents a summary of these two concepts.

	K-Means	GMM
Ideas	Unsupervised ML, separate data into K clusters based on similarity	Unsupervised ML, probabilistic model, represents data as a mixture of Gaussian distribution
Pros	- Simple and computationally efficient - Easy to understand and interpret the results	- Flexible and adapted to complex distribution - Does not assume equal variance or spherical clusters - Provides probabilities of belonging instead of pure labels
Cons	- Requires a specified number of clusters K - Sensitive to the initial placement of cluster centroids - Assume that clusters are spherical and have equal variance	- Requires a specified number of clusters K - Computationally more expensive that K-means - Requires estimating the parameters of the Gaussian distributions (mean and std) - May converge to local optima

Table 18: K-Means and GMM pros and cons.

To enhance method diversification and gain deeper insights into the scenario, we can explore various datasets using our algorithms. Initially, we can work solely with the depth points. With a frame size of 640×480 pixels, we have 307,200 data points with one feature: depth. However, we recognize that points belonging to the same cluster are spatially close as they constitute the same object. To incorporate this knowledge, we can augment the dataset by adding the row and column of the pixel as additional features.

Since our dataset encompasses diverse features, ensuring equitable significance among these features is crucial. Normalizing the data guarantees that no particular feature carries undue influence over the others. In instances where a specific feature, such as depth, exhibits a superior clustering representation during algorithm optimization, we retain the flexibility to manually enhance its weight. We need to clean our dataset to some extent. As mentioned earlier, a depth pixel with a value of 0 indicates that nothing has been detected at that location. However, this could be due to the object being either too far or too close to the camera. Since we cannot determine the exact reason, we should exclude such data points. Additionally, there are instances when the camera detects points that are excessively far from the camera (values > 30000). These outliers should also be removed.

It's important to remember that clustering algorithms can be computationally demand-

ing and might not be suitable for on-board processing. Therefore, we must explore alternative mid-range localization techniques.

b) Background Removal

The challenge with the clustering approach we just discussed lies in cluster separation. The objective of this section is to explore a manual method for achieving separation without relying on resource-intensive algorithms.

Eliminating the first cluster located at distance d_1 on **Figure 38** is straightforward. As d_1 remains constant over time, being attached to the mechanism fixed to the drone, we can exclude all depths below d_1. Removing the background is more complex due to the changing altitude of both drones, causing d_2 and d_3 to vary. However, there exists a simple approach to distinguish these clusters. With access to both drones' altitudes, we can discard depths exceeding a predefined distance. Calculating this distance can be achieved using various methods (**Table 19**).

Descriptions	**Expressions**	**Numerical values** ($z_{sd} = 5m, z_{td} = 3m$)
Mean distance between the TD and the floor.	$d = z_{sd} - 0.5 \times z_{td}$	$d = 5 - 0.5 \times 3 = 3.5m$
Altitude of the SD minus a safety margin.	$d = z_{sd} - d_{margin}$	$d = 5 - 1 = 4m$
Percentage of the SD altitude.	$d = w \times z_{sd}$	$d = 0.95 \times 5 = 4.5m$

Table 19: z_x is the altitude of drone x and d_x is the distance from the SD to object x.

The effectiveness of each discarding method will be evaluated subsequently.

After isolating the data points around d_2, we can calculate their 3D point cloud. To determine the centroid of this resulting point cloud, we can straightforwardly calculate the average of these 3D coordinates.

In both of these algorithms, only the TD's position is measured, not its orientation. While other algorithms could ascertain the aircraft's orientation using the point cloud's principal components (representing wing and body axes), such methods would introduce complexity and latency. Furthermore, considering that the TD will likely approach the SD with a reasonably accurate orientation, and that ArUco markers can determine orientation at close range, the choice was made to exclusively measure position using depth.

4.1.4 Extended Kalman Filter

With multiple measurement sources available, our goal is to combine them to estimate the state of the TD. To achieve this, we employ an Extended Kalman Filter (EKF). The EKF is a recursive estimation algorithm that uses a set of equations to estimate the true state of a dynamic system. It's particularly useful when dealing with nonlinear systems and uncertain measurements.

To accurately track the position and orientation of the TD, we will represent the orientation using quaternions. The use of quaternions allows for an efficient and compact representation of three-dimensional rotations, avoiding the problems associated with other representations like Euler angles. However, representing the dynamics of the TD using quaternions results in a non-linear system. To handle this non-linearity and improve the estimation accuracy, we will employ an Extended Kalman Filter (EKF). The EKF is a variant of the Kalman Filter that can handle non-linear systems by linearizing the system dynamics around the current estimate. The equation of the process dynamics is:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ y_k &= h(x_k) \end{aligned} \tag{39}$$

where f is a nonlinear function of the state (x_k) and input (u_k), for which we need to look at the continuous-time dynamics. y_k represents the measurement model and h is a function that maps the state variable x_k to the expected measurement values. The measurement model is expressed as:

$$h(x_k) = H_k x_k \tag{40}$$

To facilitate clear comprehension, it's important to establish various reference frames:

- **Inertial Frame(I)**: This can be visualized as a local tangent plane to the planet's surface.
- **Shuttle Body Frame (SD)**: This frame indicates the shuttle's position and orientation concerning the inertial frame, denoted as $[p^T_{k,shuttle}; q^T_{k,shuttle}]$.
- **Target Body Frame (TD)**: Similarly, this frame denotes the target's position and orientation relative to the inertial frame, expressed as $[p^T_{k,target}; q^T_{k,target}]$
- **D455 Camera Frame (C)**: This refers to the frame from which the ArUco pose measurements and depth measurements originate. These measurements need to be preconverted into the SD body frame by utilizing the corresponding translation vector and rotation matrix between the two frames: $t_{C,SD}; R_{C,SD}$

The state vector, denoted as $x_k = [p_k^T; v_k^T; q_k^T]^T$, comprises three components: p, representing the relative position between vehicles expressed at the shuttle in $\mathbb{R}^3$; v, representing the

linear relative velocity expressed at the shuttle in $\mathbb{R}^3$; and q, a unit quaternion describing the relative orientation between vehicles expressed at the shuttle in $\mathbb{R}^4$. For the sake of simplicity, we adopt a model with constant velocities. For the linear dynamics, we then have:

$$\begin{aligned} \dot{p} &= v \\ \dot{v} &= 0 \end{aligned} \tag{41}$$

Considering the rotational dynamics, we can make the following assumption:

$$\dot{q} = \frac{1}{2}\Omega(\omega_k)q \tag{42}$$

where ω_m represents the measured angular velocity of the shuttle. To simplify the assumption, we consider the angular velocity of the target to be zero, making ω_m an input (u_k) for the EKF calculation. The function $\Omega_m(\omega_m)$ is skew-symmetric matrix defined[17] using:

$$\Omega(\omega_k) = \begin{bmatrix} 0 & -\omega_{k,x} & -\omega_{k,y} & -\omega_{k,z} \\ \omega_{k,x} & 0 & \omega_{k,z} & -\omega_{k,y} \\ \omega_{k,y} & -\omega_{k,z} & 0 & \omega_{k,x} \\ \omega_{k,z} & \omega_{k,y} & -\omega_{k,x} & 0 \end{bmatrix} \tag{43}$$

Discretizing the linear part is a straightforward process:

$$\begin{aligned} p_{k+1} &= p_k + dt\dot{p} = p_k + dtv \\ v_{k+1} &= v_k + dt\dot{v} = v_k \end{aligned} \tag{44}$$

On the other hand, the rotational part presents more complexity. To address this, we will make the following assumption:

$$q_{k+1} = \exp\left(\frac{1}{2}\Omega(\omega_k)dt\right)q_k \tag{45}$$

To implement the EKF, we just need to compute the derivative of $f(x_k, u_k)$:

$$\frac{\partial f(x_k, u_k)}{\partial x_k} = A_k(\omega_k) = \begin{bmatrix} I_3 & dt \times I_3 & 0_{3\times4} \\ 0_{3\times3} & I_3 & 0_{3\times4} \\ 0_{4\times3} & 0_{4\times3} & exp\left(\frac{1}{2}\Omega(\omega_k)dt\right) \end{bmatrix} \tag{46}$$

The function f is then :

$$f(x_k, u_k) = A_k(\omega_k)x_k = F_K x_k \tag{47}$$

Regarding the measurements, the equations will depend on the measurement source. Since the objective is to determine the position of the TD from the viewpoint of the SD, it is essential to represent all measurements within the reference frame of the SD body.

Using GPS measurements, we can determine the relative position of the TD with respect to the SD. Utilizing the transmission of values from the TD to the SD, we can also establish a relative quaternion between the two drones. Given that the reference frame of the GPS corresponds to the inertial frame, it becomes necessary to transform the measurements into the shuttle body frame.

$$\begin{aligned} p_{k,GPS} &= R(q_{k,shuttle})(p_{k,target} - p_{k,shuttle}) \\ q_{k,GPS} &= q_{k,shuttle}^{-1} \otimes q_{k,target} \\ y_{k,GPS} &= \begin{bmatrix} p_{k,GPS} \\ q_{k,GPS} \end{bmatrix} \\ H_{k,GPS} &= \begin{bmatrix} I_3 & 0_{3\times3} & 0_{3\times4} \\ 0_{4\times3} & 0_{4\times3} & I_4 \end{bmatrix} \end{aligned} \tag{48}$$

For ArUco measurements, the defined measurements expressed in the SD body frame and transition matrix are :

$$\begin{aligned} y_{k,aruco} &= \begin{bmatrix} p_{k,aruco} \\ q_{k,aruco} \end{bmatrix} \\ H_{k,aruco} &= \begin{bmatrix} I_3 & 0_{3\times3} & 0_{3\times4} \\ 0_{4\times3} & 0_{4\times3} & I_4 \end{bmatrix} \end{aligned} \tag{49}$$

And finally, for depth measurements also expressed in the SD body frame :

$$\begin{aligned} y_{k,depth} &= p_{k,depth} \\ H_{k,depth} &= \begin{bmatrix} I_3 & 0_{3\times3} & 0_{3\times4} \end{bmatrix} \end{aligned} \tag{50}$$

Although it is not implemented in this thesis, attitude estimation is also possible using depth measurements. The equations would need to be modified accordingly if added.

The state equation and the measurement equations can be written with their respective noise:

$$\begin{aligned} x_{k+1} &= A_k(\omega_k)x_k + w_{x_k} \\ y_k &= H_k x_k + w_{y_k} \end{aligned} \tag{51}$$

We assume all random variables to have a normal distribution with a mean of zero and a known covariance :

$$\begin{aligned} w_{x_k} &\sim \mathcal{N}(0, Q_k) \\ w_{y_k} &\sim \mathcal{N}(0, R_k) \end{aligned} \tag{52}$$

Ultimately, the equations of the Extended Kalman Filter (EKF) are given by[26]:

- **Predict step**
  Predicted state estimate :
  $$\hat{x}_{k+1|k} = A_k(\omega_k)x_{k|k} \tag{53}$$
  Predicted covariance estimate :
  $$P_{k+1|k} = F_k P_{k|k} F_k^T + Q_k \tag{54}$$
  where Q_k is the covariance matrix of the process noise.

- **Update step**
  Measurement residual :
  $$s_{k+1} = y_{k+1} - H_{k+1}\hat{x}_{k+1|k} \tag{55}$$
  where y_k is the measurement and H_k is the transition matrix.
  Residual covariance :
  $$S_{k+1} = H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1} \tag{56}$$
  where R_k is the covariance matrix of the measurement noise.
  Near-optimal Kalman gain :
  $$K_{k+1} = P_{k+1|k} H_{k+1}^T S_{k+1}^{-1} \tag{57}$$
  Updated state estimate :
  $$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1} s_{k+1} \tag{58}$$
  Updated covariance estimate :
  $$P_{k+1|k+1} = (I - K_{k+1} H_{k+1}) P_{k+1|k} \tag{59}$$

To combine various measurements, a ROS2 node named *sensor_fusing_node* was developed. It subscribes to topics containing measurements from different sources and updates the EKF's state accordingly. Adjustments are made to the transition matrix H and the covariance matrix of the measurement noise R to account for varying measurement precision and the number of states measured. The covariance matrix of the process noise Q can also be adjusted. Lower values on the diagonal of the Q matrix increase trust in predictions. If a specific measurement source is more accurate, the values in its R matrix diagonal should be reduced to enhance trust. This approach allows the fusion of measurements from ArUco boards, depth measurements, and GPS positions simultaneously.

4.1.5 State Machine

The operation of the entire catching phase can be defined by the state machine diagram in **Figure 39**.

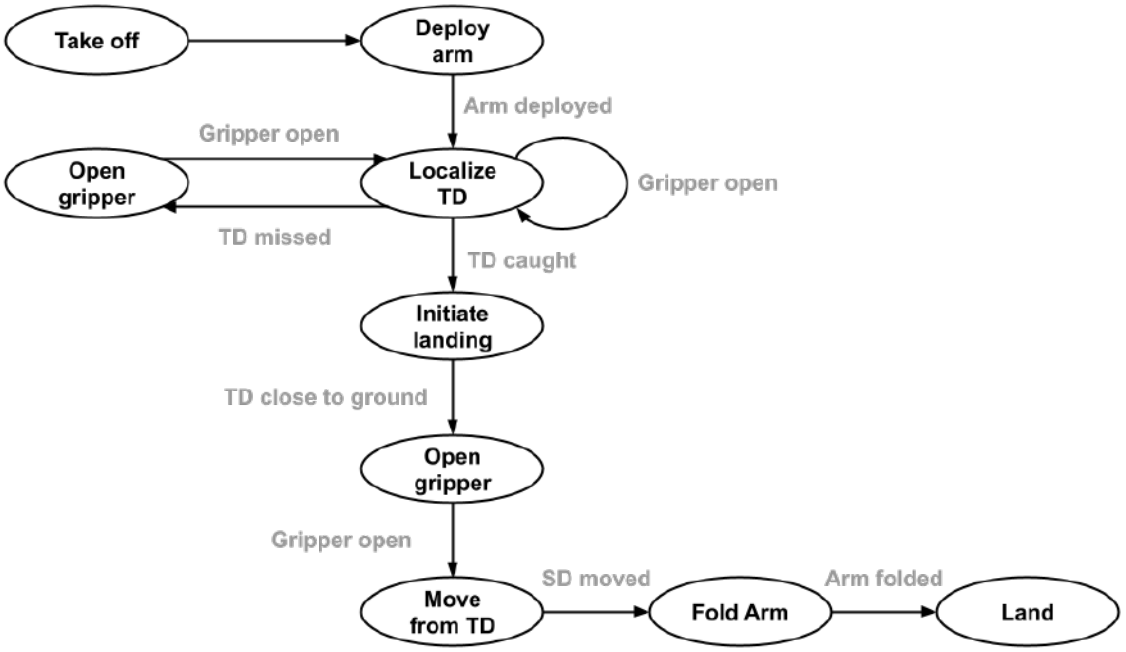


Figure 39: State Machine of the operations.

This state machine always begins in the take-off phase and concludes in the landing phase. After the shuttle drone has taken off, it deploys its arm and awaits confirmation. Following this, it enters the TD localization phase in which it endeavors to approach the TD. If the gripper closes but fails to capture the TD, it reopens the gripper and continues the localization process. Once the TD is successfully captured, the SD initiates the landing procedure by descending close to the ground. It subsequently opens the gripper to release the TD and moves away. Once a safe distance from the TD is achieved, it retracts its arm and proceeds to land.

4.2 Results and Analysis

In this section, we will examine and evaluate the effectiveness of our detection algorithms. The majority of the tests outlined here were conducted indoors within our laboratory. Unfortunately, real flight tests could not be carried out due to scheduling constraints. Actual field trials were not feasible during the final month of this study, as the algorithms were not prepared for execution before that period. As a result, the results presented will be confined to this indoor analysis.

4.2.1 ArUco Markers

In this subsection, we will examine the effectiveness of ArUco marker tracking. The initial step involves determining the detection accuracy of a single marker for each set of expressed coordinates. To ascertain this accuracy, we position a 14.8cm-side marker at known locations along each axis, enabling us to assess the precision of the ArUco tracking for each axis. This approach enables us to establish that the accuracy of ArUco markers is contingent on factors such as the camera's distance, the marker's size, and its placement relative to the frame's center. For instance, a marker positioned at the center of the frame would exhibit a precision of less than 5% at 1m (z-axis) and less than 7% at 3m (z-axis). In contrast, a marker situated more towards the frame's periphery (1m along the x-axis from the center) would yield a precision of less than 8% at 1m (z-axis) and 9% at 3m (z-axis). Determining the accuracy of ArUco tracking for rotation is more challenging. Nevertheless, utilizing the same setup, we were able to ascertain a precision consistently below 3 degrees.

It's important to acknowledge that our setup is relatively basic and doesn't facilitate accurate measurement of this precision. Additionally, the presence of multiple markers on the TD and the utilization of the Extended Kalman Filter (EKF) are expected to enhance this accuracy. A more detailed approach to measuring accuracy was undertaken by Francisco Azevedo during his master's thesis[**azevedo_francisco_nodate**].

The marker size significantly influences detection. Larger markers result in an expanded detection range. To achieve this, we altered the configuration in **Figure 36** to a simpler layout, employing only one large marker on each board. Marker sizes were set at 14.8cm for central markers and 8.9cm for outer markers. This modification aimed to maximize the markers' detection range.

Figure 40: New configuration of ArUco boards on the target drone.

The subsequent phase of our research involves assessing the performance of our ArUco tracking algorithms. To achieve this, we captured a video[27] of the TD with the ArUco boards in our laboratory, varying the distance and rotations. We then applied the tracking algorithms to the video in real-time.

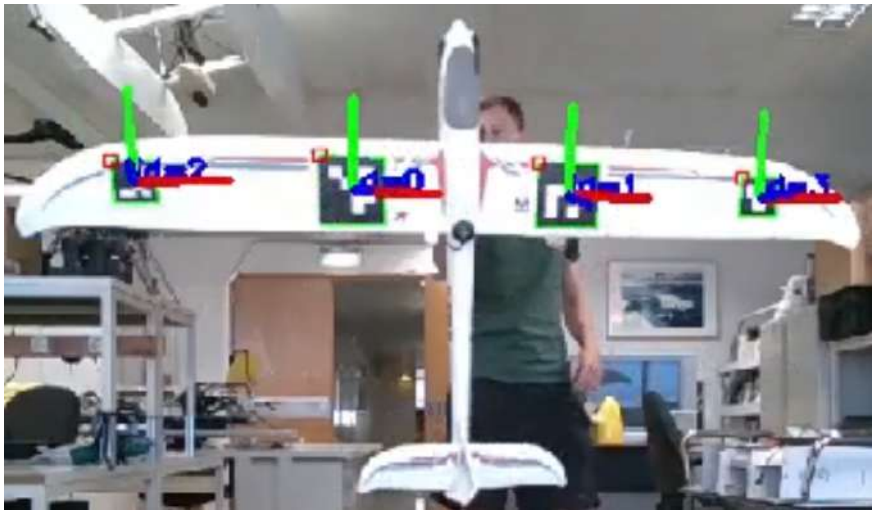


Figure 41: Exemple of detection of all 4 ArUco boards.

The offsets referred to in **Section 4.1.2** is not apparent in **Figure 41**. These offsets are used to translate all board positions to a common central point. The tracking results are as in **Figure 42**.

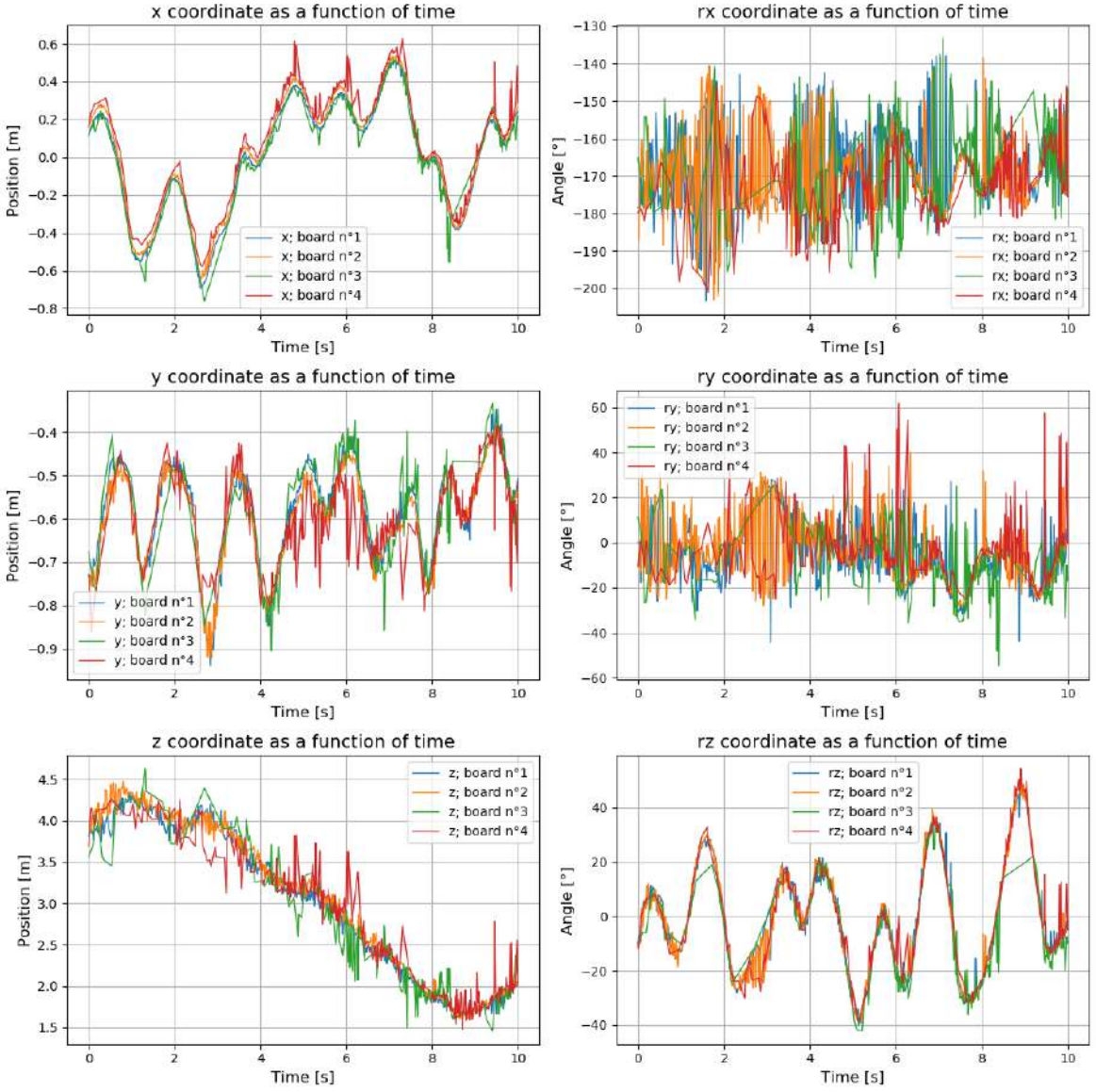


Figure 42: Tracking of the ArUco boards positions and estimations.

As evident, the signals can exhibit noise, particularly for the roll and pitch (rx and ry). The positive aspect is that the most crucial rotation, the yaw, is the least noisy. Observing the z coordinate, it's apparent that the tracking appears less shaky at closer ranges. Given that the ArUcos are employed for close-range tracking, we can perform a similar analysis from a

shorter distance (**Figure 43**).

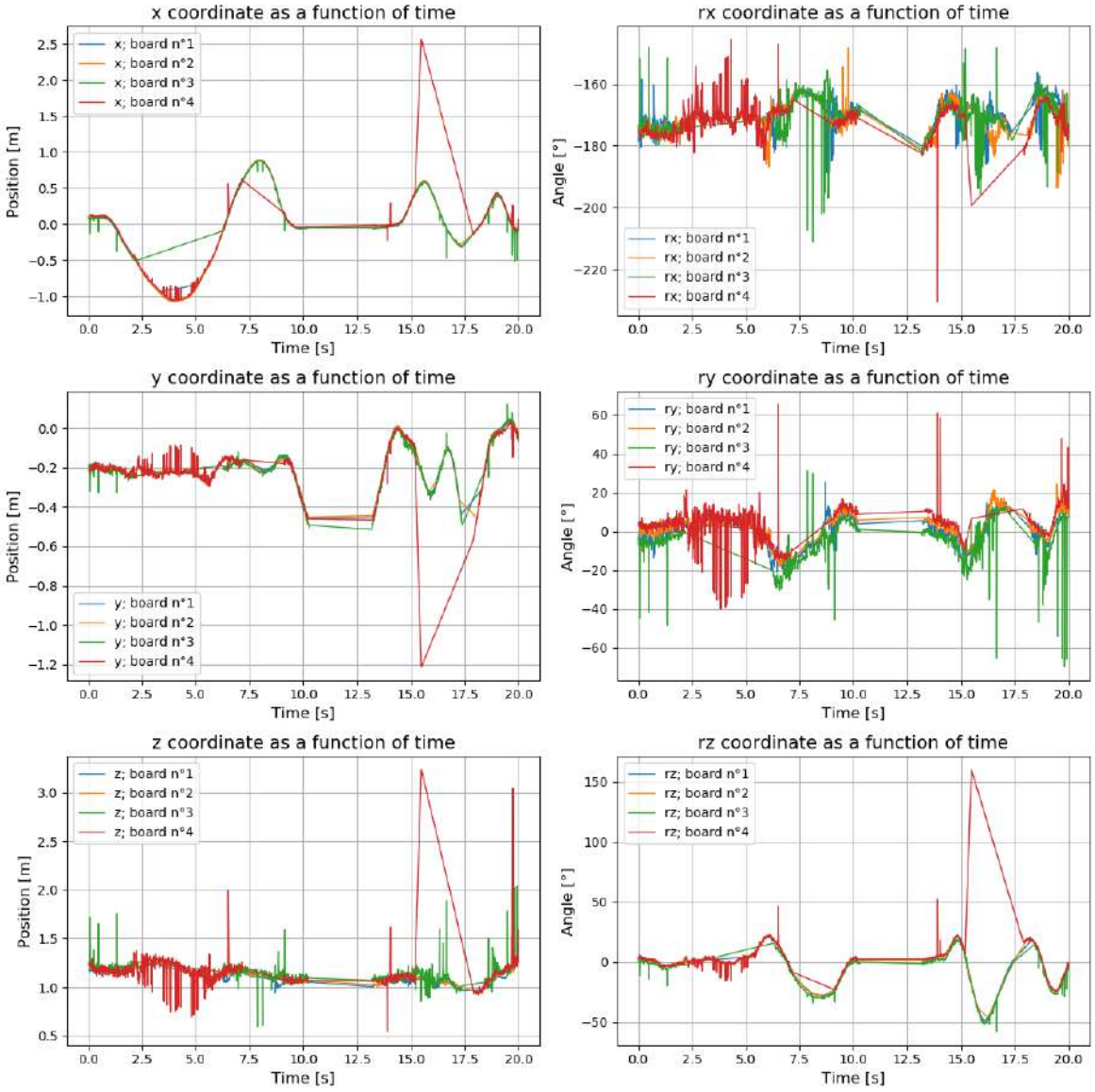


Figure 43: Tracking of the ArUco boards at close range.

The first noticeable aspect is the improved smoothness of the signals. They exhibit significantly less noise compared to the previous instance. This can be attributed to the closer distance and potentially the fact that the TD was less shaky during this recording. The yaw rotation still appears as the smoothest rotation in this graph. Upon inspecting the signals of each board, it appears that signals 3 and 4 are more erratic than signals 1 and 2. To verify this observation, we can calculate statistics for each signal (**Table 20**).

Board ID		x [m]	y [m]	z [m]	rx [°]	ry [°]	rz [°]	time diff [s]
1	$\overline{\mu}$	0.047	-0.179	1.100	-169.8	-0.874	-5.7	0.024
	σ	0.426	0.098	0.063	5.0	7.9	16.4	0.124
2	$\overline{\mu}$	-0.091	-0.182	1.125	-170.9	1.4	-3.5	0.022
	σ	0.518	0.092	0.068	4.8	7.5	14.4	0.106
3	$\overline{\mu}$	0.172	-0.188	1.145	-170.5	-5.6	-10.9	0.029
	σ	0.323	0.112	0.117	6.7	11.5	16.3	0.192
4	$\overline{\mu}$	-0.205	-0.191	1.143	-171.7	3.5	2.5	0.028
	σ	0.460	0.114	0.155	5.5	10.3	12.2	0.160

Table 20: Performance Statistics of ArUco Marker Tracking.

Contrary to our expectations, the standard deviations of boards 3 and 4 are not notably different from those of boards 1 and 2. We must explore alternative methods to validate our hypothesis. One possible approach is to conduct a frequency analysis of all signals.

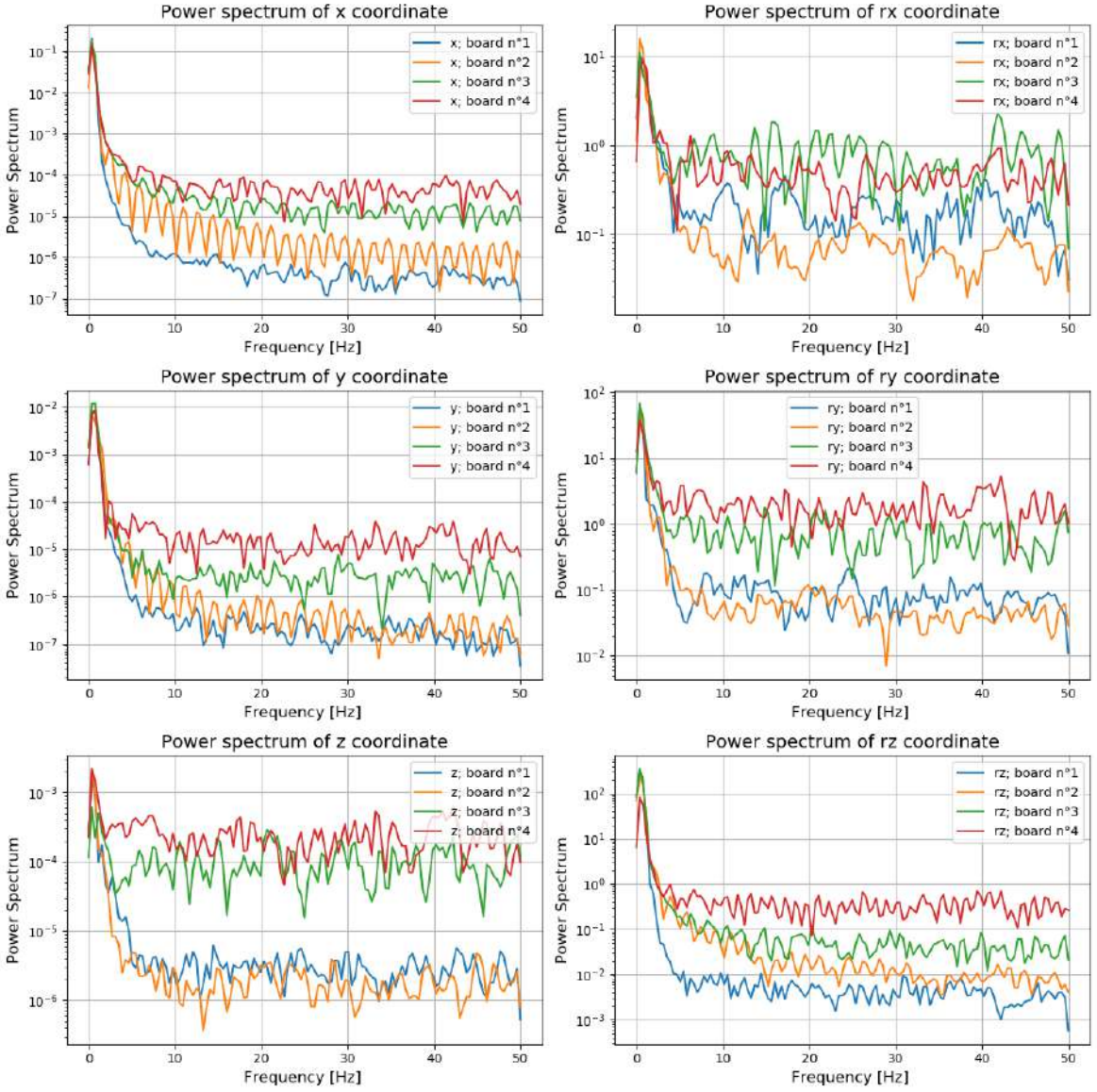


Figure 44: Power Spectrum of each coordinate for all 4 boards.

Figure 44 vividly illustrates that higher frequencies are more prevalent in signals 3 and 4. This observation indicates a higher level of noise in these signals. Several explanations can account for this phenomenon. Firstly, it can be attributed to the size of the markers. Since the smaller markers are less detectable, this is corroborated by the information in the last column of **Table 20**. It is evident that the average time between two consecutive marker detections is greater for these smaller markers, with a corresponding increase in standard deviation. Consequently, the two outer markers experience fewer detections compared to the inner markers. Another potential explanation is linked to the markers' positions on the TD.

Positioned farther from the plane's center, these outer markers necessitate larger position offsets. However, given the inherent noise in orientation data, a greater offset equates to a higher level of noise in the 3D positions as well.

One last factor to consider in explaining the noise across all four signals could be the high detection frequency. This is linked to the camera's frame rate, which currently operates at 60 frames per second (fps). However, it's worth noting that this rate might be excessive and potentially needs to be reduced to facilitate on-board computations.

To achieve improved accuracy in orientation tracking, it's worth considering alternative configurations of ArUco boards. Exploring the possibility of using a single board with multiple large ArUco markers could potentially enhance the stability of orientation detection. However, due to time constraints, this concept won't be explored within the scope of this thesis.

Despite the absence of outdoor tests, we managed to gather data from a single flight attempt (**Figure 45**). Regrettably, the recorded depth image was of poor quality and is not usable for analysis. The field trial was conducted with the SD flying above the TD, which was mounted on tripods approximately 1.50 meters high, as the control algorithms were still in development. The camera was fixed to the SD at an angle of around 20 degrees. ArUco markers were printed and affixed to the TD's wings, and the weather was very sunny. These conditions collectively contributed to poor performance of the detection algorithms in the recorded video. The reflective nature of the black ink on the markers posed a challenge for detection. Additionally, the camera's angle and the fact that the SD hovered about 3 meters above the TD resulted in a small marker surface for detection. These limitations underscore the necessity for more comprehensive outdoor testing.

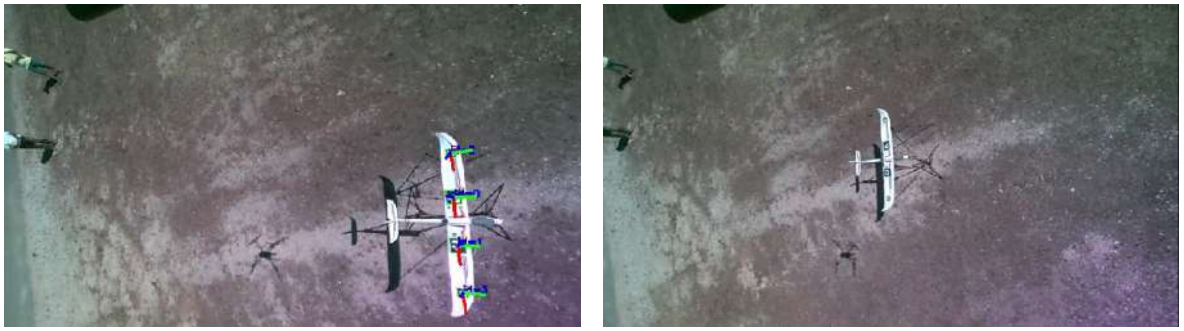


Figure 45: Screenshots from the video recorded during field trials.

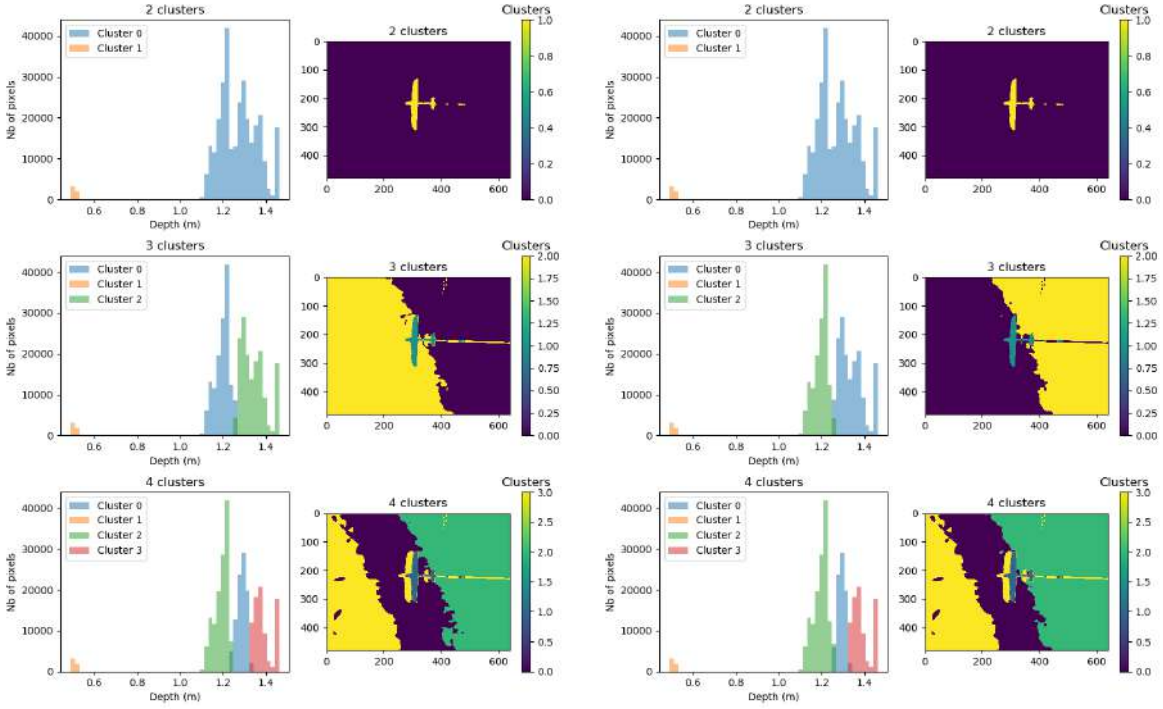
4.2.2 Depth Detection

a) Depth Clustering

In this subsection, we will examine the outcomes of the depth clustering algorithms. As previously elucidated, external testing was unfeasible during this summer. To assess the depth algorithm, an indoor setup was developed. However, emulating the flight conditions, particularly the clear differentiation between presumed clusters, posed the most significant challenge. Indoors, the background is not as distant as it would be during flight. Furthermore, the floor and ceiling are within the camera's field of view. Consequently, the outcomes may not be as consistent as anticipated. To replicate an aerial perspective of the plane, a cardboard replica affixed to a surface was employed (**Figure 46**).

Figure 46: Cardboard replica of the TD.

The initial phase of testing our clustering algorithms involves their use on photos. Images of the replica were captured from various distances and angles. Initially, the clustering was evaluated using only one feature: depth. The outcomes for both GMM and K-Means were shown in **Figure 47**.

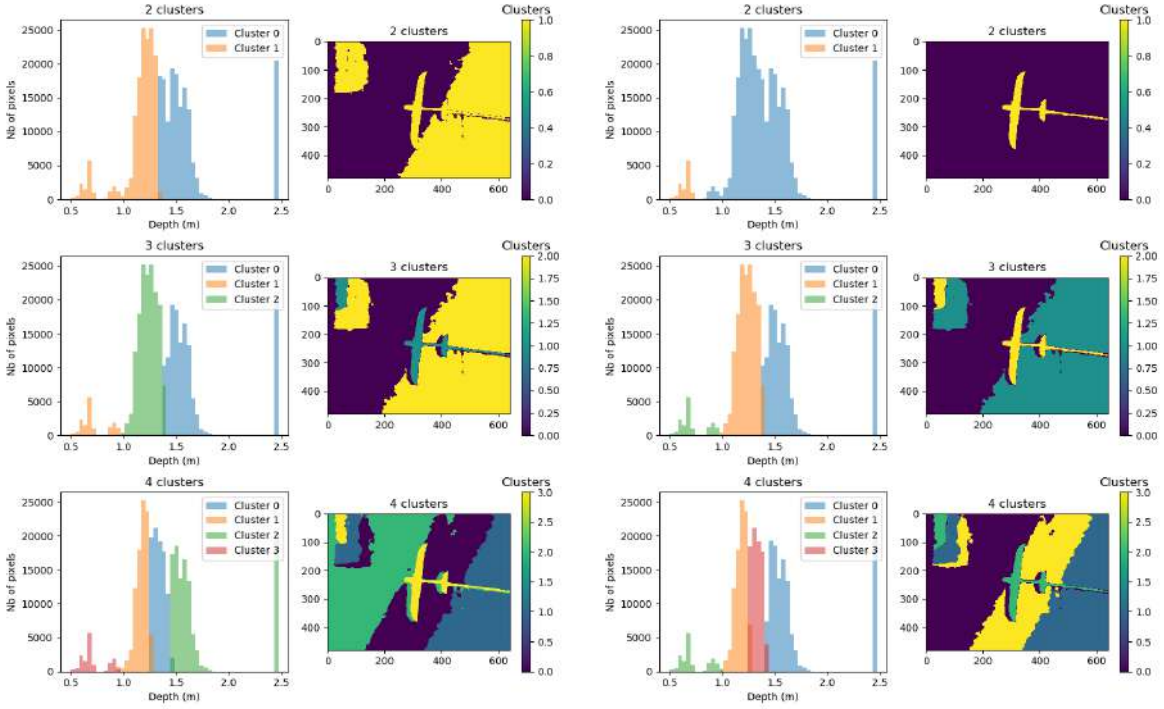


(a) K-Means Clustering.

(b) GMM Clustering.

Figure 47: Clustering using only the depth as a feature with satisfying results.

As depicted in **Figure 47**, there is a clear separation in the depth distribution between the TD and the background, leading to effective clustering for both methods. The specific number of clusters isn't highly relevant, given that the TD consistently stands out from the background. However, it's important to recognize that this might not be the case in all scenarios **Figure 48**.



(a) K-Means Clustering.

(b) GMM Clustering.

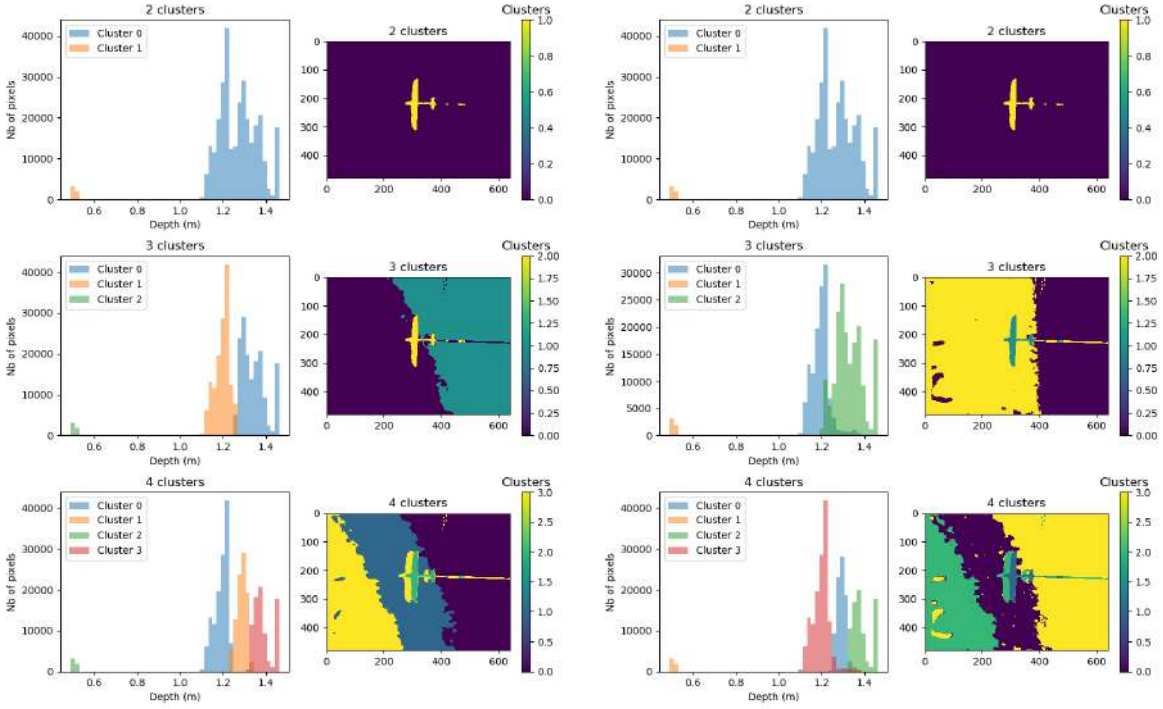
Figure 48: Clustering using only the depth as a feature with poor results.

In **Figure 48**, it's evident that the clustering performance wasn't as successful as previously observed. The separation between the TD and the background isn't as distinct, except for the GMM clustering with two clusters. This can be attributed to the less favorable depth distribution[3]. It's worth noting that the chosen number of clusters also influences the clustering outcomes, as seen with a drop in efficiency from 2 to 3 clusters in GMM clustering.

The initial conclusion drawn is that the clustering outcome is influenced by the number of defined clusters, but more prominently by the depth distribution, as anticipated. However, we anticipate a more distinct distribution for flight conditions.

The subsequent stage involves incorporating the pixel's position in the image as a feature, which encompasses the rows and columns of each depth pixel. When applied to the image from **Figure 47**, the outcomes are as shown in **Figure 49**.

[3]The peak at the greatest depth results from the data preprocessing mentioned in the methods section. It was omitted for the clustering process itself.

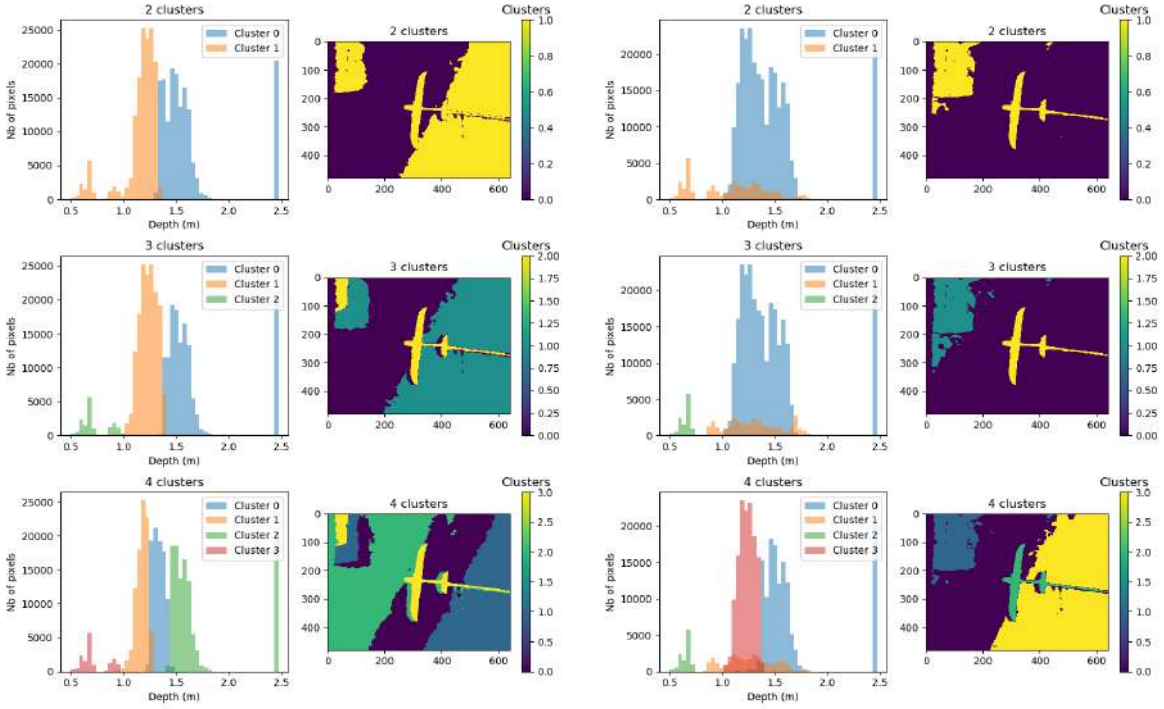


(a) K-Means Clustering.

(b) GMM Clustering.

Figure 49: Clustering using the depth and the pixel rows and columns as features with satisfying results. The weight of the depth feature is 10 times the one of the others.

Once more, the TD drone remains distinctly distinguished from the background, benefiting from a favorable depth distribution. The introduction of additional features did not change the algorithm's results. However, when considering the image from **Figure 48**, the outcomes are different (**Figure 50**).



(a) K-Means Clustering.

(b) GMM Clustering.

Figure 50: Clustering using the depth and the pixel rows and columns as features with poor results. The weight of the depth feature is 10 times the one of the others.

In this instance, the inclusion of spatial features appears to reduce the efficacy of the clustering. Particularly with GMM, the clustering becomes more dispersed and doesn't remain concentrated solely on the depth distribution, even with a weight that's tenfold more significant than the other features. These outcomes underscore the pivotal role of the depth distribution in achieving desirable results. Moreover, they demonstrate that the incorporation of spatial features doesn't inherently enhance the outcomes. Instead, it adds to the computational time and complexity of the algorithms. Consequently, the decision was made to exclusively employ depth for this particular method.

To ascertain the centroid of a cluster, we can compute the average position of all the points within that cluster (**Figure 51**). However, identifying which cluster corresponds to the TD is not always straightforward. In our experiments, the TD is typically found in the nearest cluster to the camera. Yet, under real operational conditions, the catching mechanism might result in a closer cluster being formed.

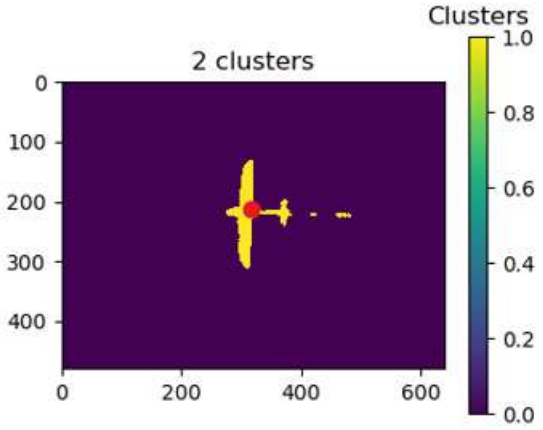


Figure 51: Cluster Centroid.

The ultimate stage of testing the clustering algorithm involves its real-time implementation. As anticipated, the algorithms prove to be quite resource-intensive, resulting in substantial intervals between two consecutive detections (**Table 21**).

	Time between two detections [s]
$\overline{\mu}$	0.918
σ	0.467

Table 21: Mean and Standard Deviation of the time between two consecutive datapoints.

The video[27] analysis also reveals another issue. When the TD is not within the camera's FOV, there is no intermediate cluster. However, due to the fixed number of clusters, the algorithm will attempt to identify differentiation where it shouldn't exist. This phenomenon is evident in **Figure 52** where the background is divided in two clusters.

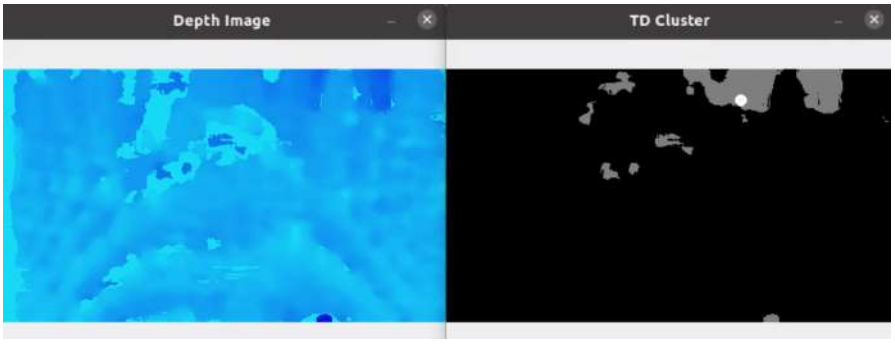


Figure 52: Clustering algorithm works poorly when the TD is not in the FOV of the camera.

Despite these suboptimal outcomes, the algorithm performs well when the TD is detected. In **Figure 53**, the white dot represents the mean of all datapoints in the grey cluster projected onto the image. While we lack a precise measure of the accuracy of this method, it can be deemed visually satisfactory.

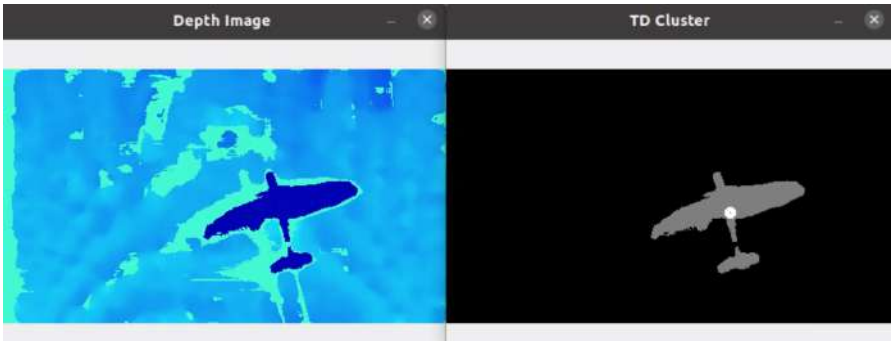


Figure 53: Clustering algorithm works when the TD is in the FOV of the camera.

Considering the array of pros and cons associated with the clustering technique for TD localization, the decision was made to refrain from utilizing these algorithms. Despite the encouraging outcomes, the low detection frequency (1HZ) and, more significantly, the issues stemming from the fixed number of clusters render this method insufficiently dependable. Consequently, we must explore alternative solutions for detecting the TD at mid-range.

b) Background Removal

In this section, our exploration is significantly constrained by the unavailability of real flight testing opportunities. The functionality of our algorithms, as presented in **Section 4.1.3**, relies on two critical conditions: maintaining a specific altitude and ensuring a distinct separation between the target drone and the background. Regrettably, our lab setting does not fulfill either of these conditions. To elaborate, although we can achieve a clear differentiation between the TD and the background at close range (1m), this distinction becomes muddled beyond this proximity due to the depth camera detecting the lab floor. This interference persists even when conducting outdoor tests. To partially mitigate this issue, we can attempt to simulate the desired environment by positioning the TD roughly 1 meter away from the camera and discarding depth points beyond 1.30 meters and below 0.60 meters.

Determining the accuracy of this position estimation method, as we did for the ArUco markers, is also more complex. This complexity arises because for an accurate position calculation, the entire TD should be within the camera's FOV. For instance, if a specific part of the TD, such as the left wing in **Figure 54**, is outside the FOV, the calculated

centroid will shift towards the right of the TD's center. This is because the centroid is computed by averaging the 3D positions of all measured datapoints.

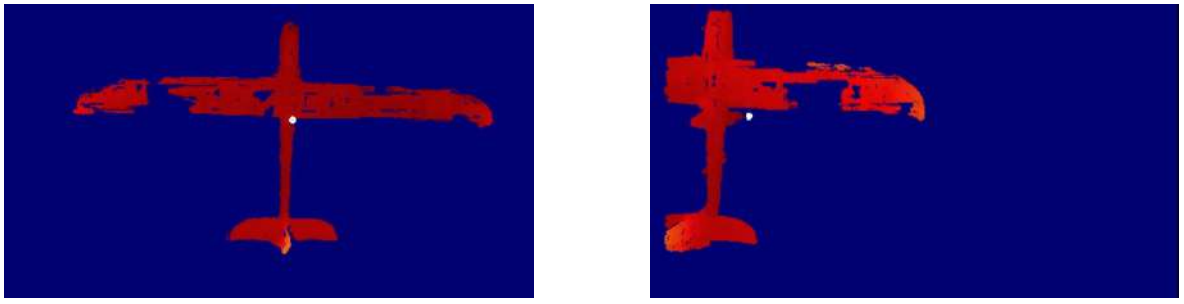


Figure 54: Calculation of the centroid when the plane is entirely within the camera's FOV and when it's not.

However, despite these challenges, we have relied on visual inspection and taken into account the camera's performance (depth accuracy $< 2\%$ at 4 meters) to approximate this precision to be around 50 cm at a distance of 4 meters.

To assess the efficiency of the tracking, we can utilize the same sequence as we did for the ArUco markers and plot the detected positions.

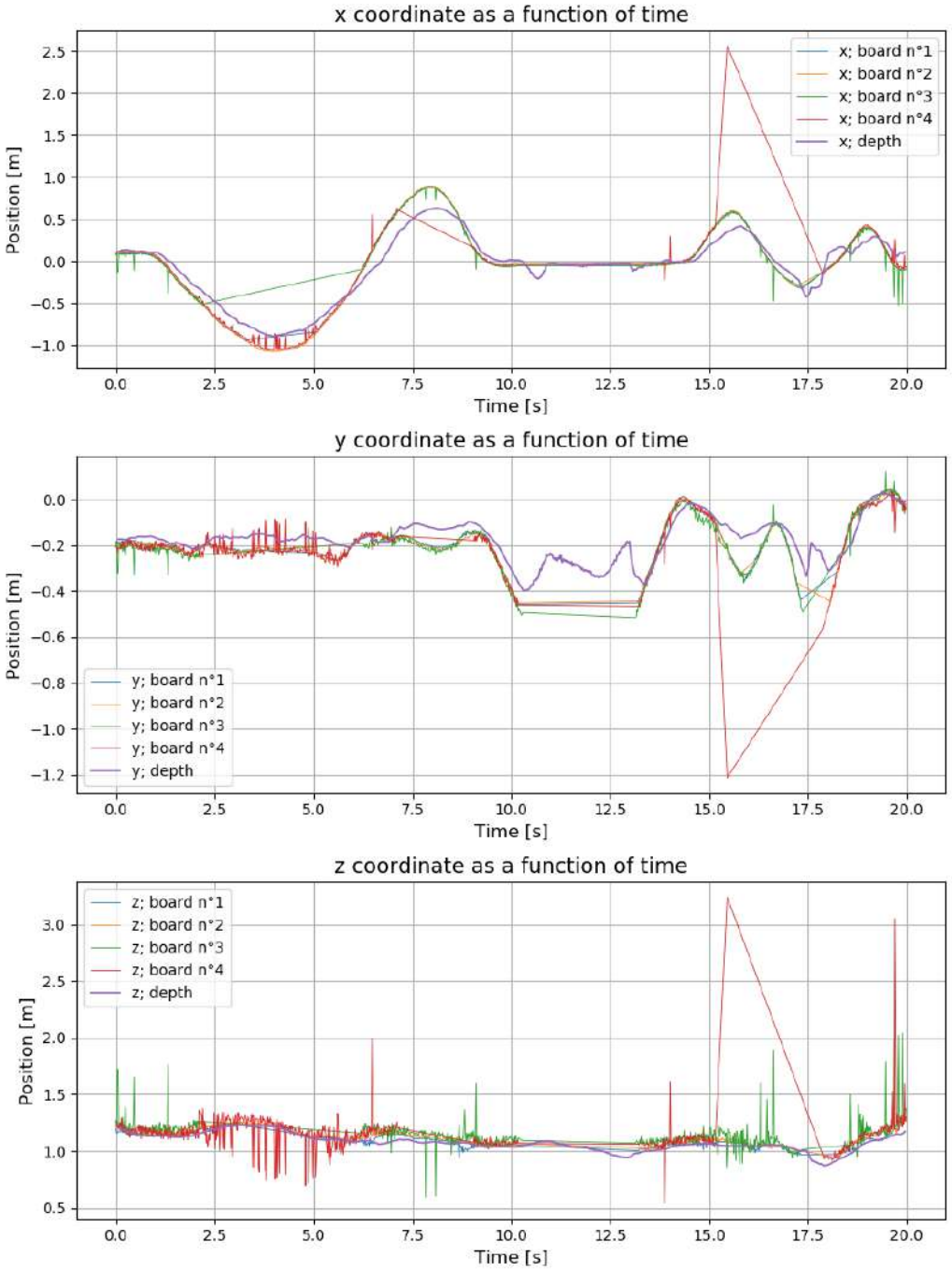


Figure 55: Tracking of the TD at close range using ArUco markers and the depth measurement.

As depicted in **Figure 55**, the tracking using depth measurements differs from that

using ArUco tags. This confirms the lower accuracy of the depth-based method. However, a significant advantage becomes apparent in the same graph, specifically between 10s and 13s. During this time interval, the markers move out of the camera's FOV. Throughout this extended period, ArUco detection is unavailable. Nevertheless, the depth measurement enables continuous detection, albeit with reduced precision. This capability enables the SD to maintain its trajectory toward re-detecting the precise markers.

Furthermore, the signal from the depth detection method exhibits considerably less noise compared to the markers detection method. This is evident both in its standard deviations and its Power Spectrum (**Table 22** and**Figure 56**).

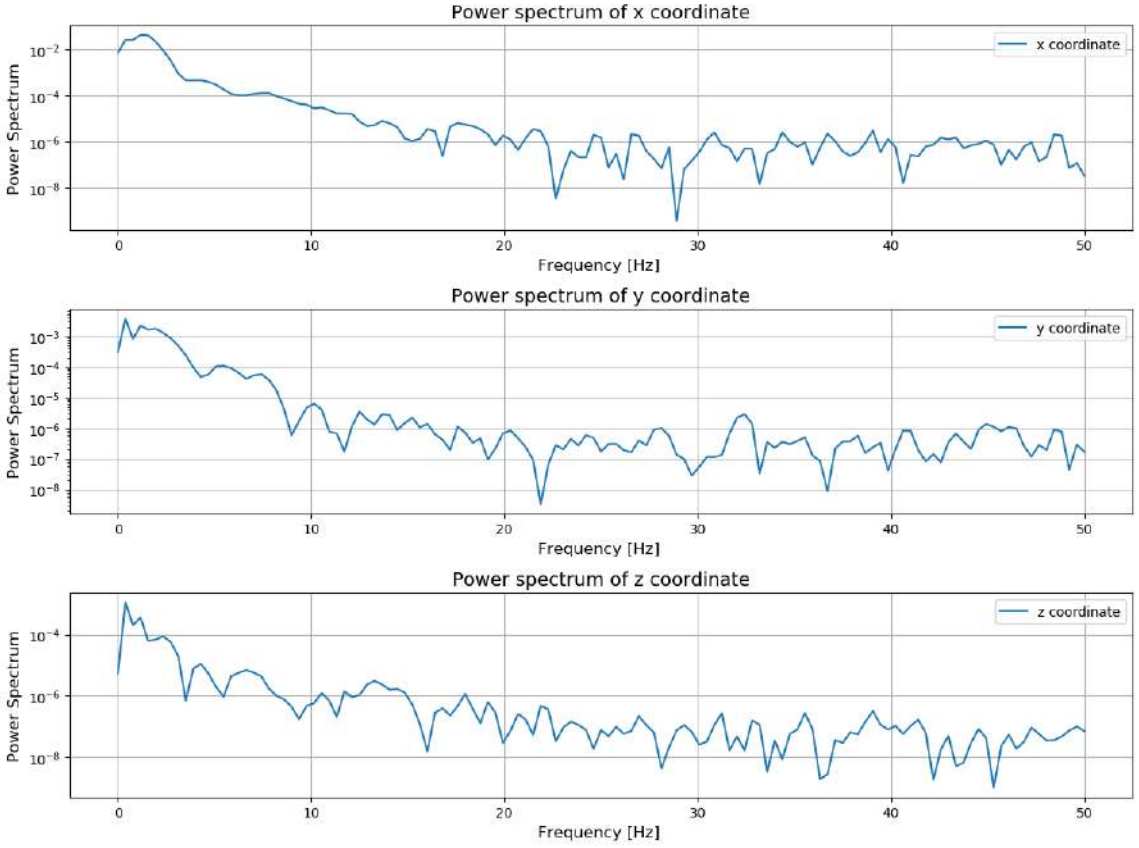


Figure 56: Power Spectrum of the 3D positions for the depth signal.

	x [m]	**y [m]**	**z [m]**	**time between frames [s]**
$\overline{\mu}$	-0.089	-0.221	1.052	0.057
σ	0.308	0.088	0.079	0.041

Table 22: Performance statistics of localization using background removal.

Table 22 highlights an interesting observation: despite the depth detection method having a lower detection frequency compared to ArUco markers, the standard deviation of the time interval between two consecutive detections is smaller. This indicates that the depth-based detection occurs more frequently and maintains a more consistent rhythm.

A video analysis[27], from which screenshots in **Figure 54** are extracted, confirms this regularity and smoothness.

4.2.3 Extended Kalman Filter

This section will provide a detailed overview of the outcomes obtained from utilizing the Extended Kalman Filter, as well as the process of fine-tuning it. As previously mentioned, our testing capabilities are somewhat constrained. Given the absence of GPS data, our focus will be on combining measurements derived from the ArUco markers and depth-based background removal. However, due to the lack of altitude information, we will establish a manual threshold distance to facilitate the removal of background depth values. Furthermore, the lack of rotational velocities for both drones as input to our filter is worth mentioning. In this section, we will set these velocities to 0. By doing so, the process model will operate under the assumption of constant orientations.

To fine-tune our filter, we will examine the same sequence mentioned earlier, as it encompasses various important scenarios: a noisy ArUco signal sequence, instances with both ArUco and depth measurements, and a sequence where only depth data is available. The initial tuning phase involves setting all covariance matrices to the identity matrix. The tracking results are as depicted in **Figure 57** and **Figure 58**.

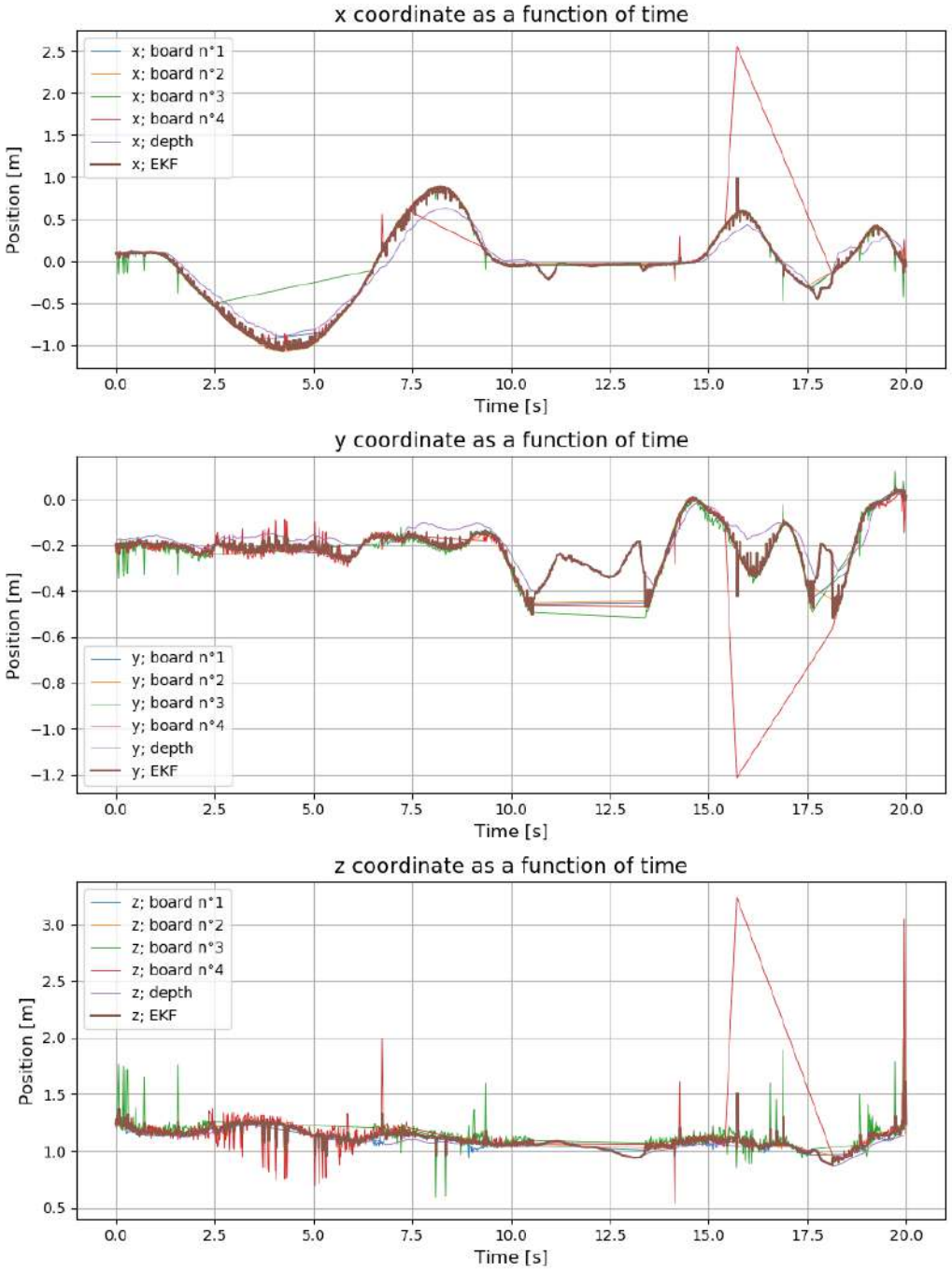


Figure 57: EKF position signals : all covariance matrices are set to identity matrices.

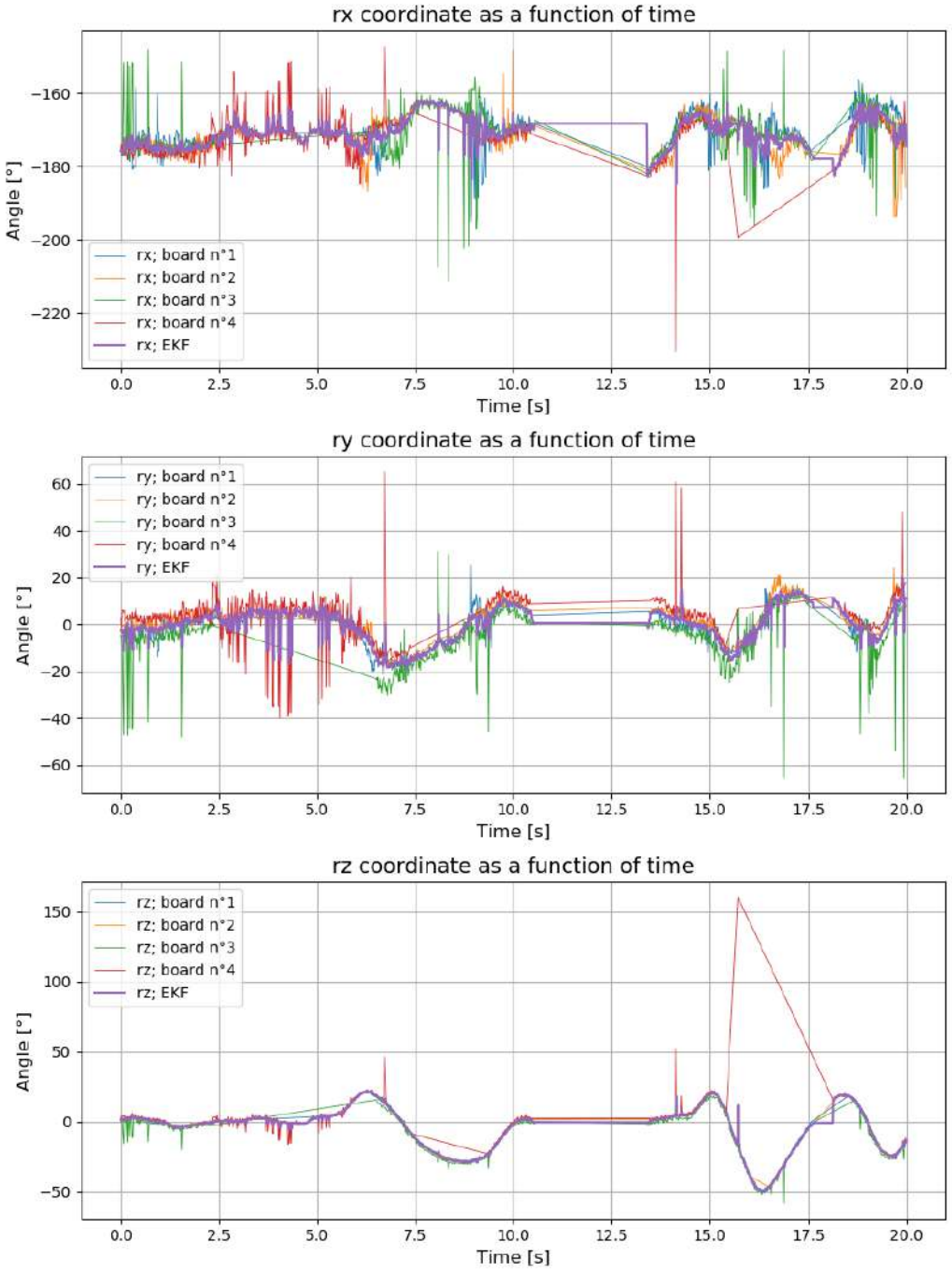


Figure 58: EKF orientation signals : all covariance matrices are set to identity matrices.

As evident from **Figure 57** and **Figure 58**, the signals from our Extended Kalman Filter (EKF) exhibit significant noise. This is expected at this stage since we haven't tuned the

filter. Let's delve into the process of tuning and explore the covariance matrices for both the model and the noise.

As explained in **Section 4.1.4**, the covariance matrix Q represents the process noise, while R represents the covariance matrix of the measurement noise. It's important to note that each measurement source possesses its own unique measurement noise, necessitating independent tuning of each R matrix. Both Q and R matrices are diagonal. By increasing the coefficients along their diagonals, we introduce more noise into their respective models. Consequently, if our intention is to place greater trust in the filter's predictions, we should decrease the corresponding coefficients. Similarly, when a measurement source proves to be accurate, lowering the coefficient in its R matrix is advisable.

With this understanding, let's examine **Figure 57** and **Figure 58**. The initial observation is that the filter heavily depends on the measurements, leading to the observed noise. To address this, we should reduce the coefficients in the Q matrix. Implementing this adjustment yields the plots in **Figure 59** and **Figure 60**.

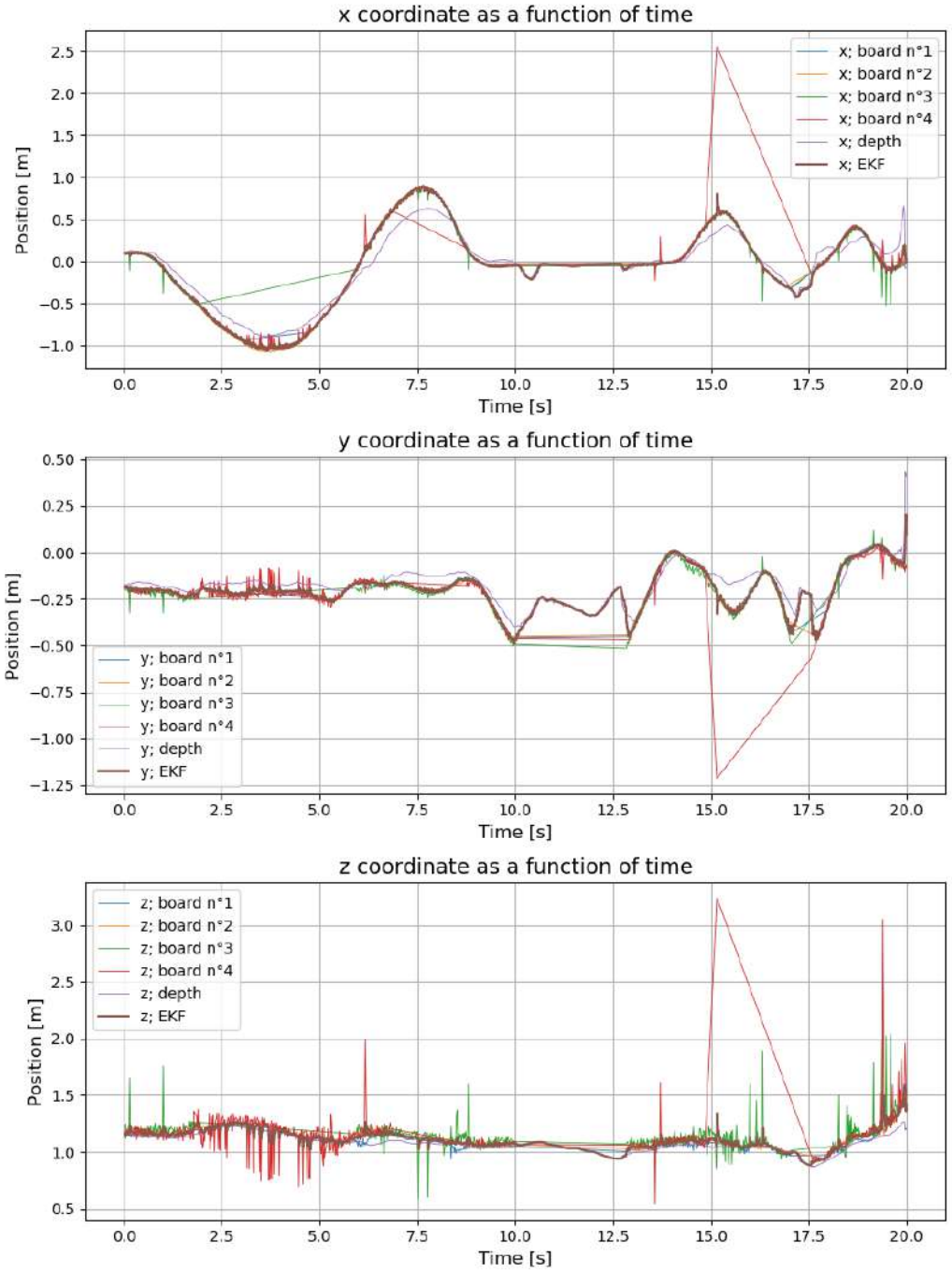


Figure 59: EKF position signals : $Q = 0.1 \times I_{10}$.

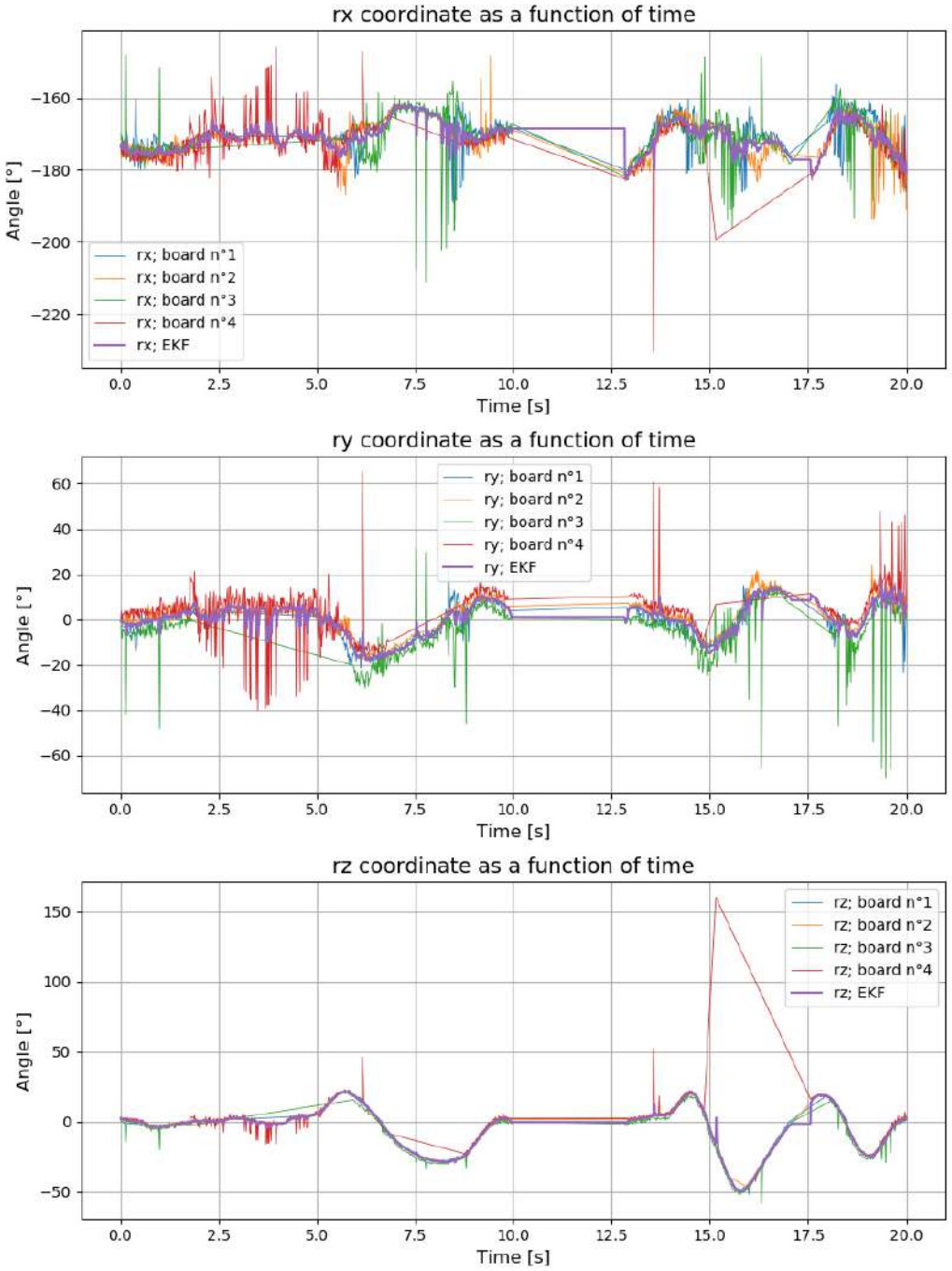


Figure 60: EKF orientation signals : $Q = 0.1 \times I_{10}$.

We observe a noticeable improvement with less noise in the signals, particularly for positions and yaw (rz). However, noise still persists in roll (rx) and pitch (ry). To address this issue,

let's attempt to fine-tune the R matrices based on the insights from **Section 4.2.1** and **Section 4.2.2**.

Considering our findings, we can conclude that boards 3 and 4 exhibit reduced detection reliability and noisier estimations. Consequently, the R coefficients for these boards should be larger compared to boards 1 and 2. The roll and pitch estimations, which are generally noisy, should also have higher coefficients.

Regarding depth measurements, adjustments are needed only for positions since we're not yet measuring orientation. Given the lower accuracy of this method compared to ArUco measurements, the R matrix coefficient should be higher.

After an iterative process of tuning and result analysis, we arrived at the pose estimation outcomes in **Figure 61** and **Figure 62**.

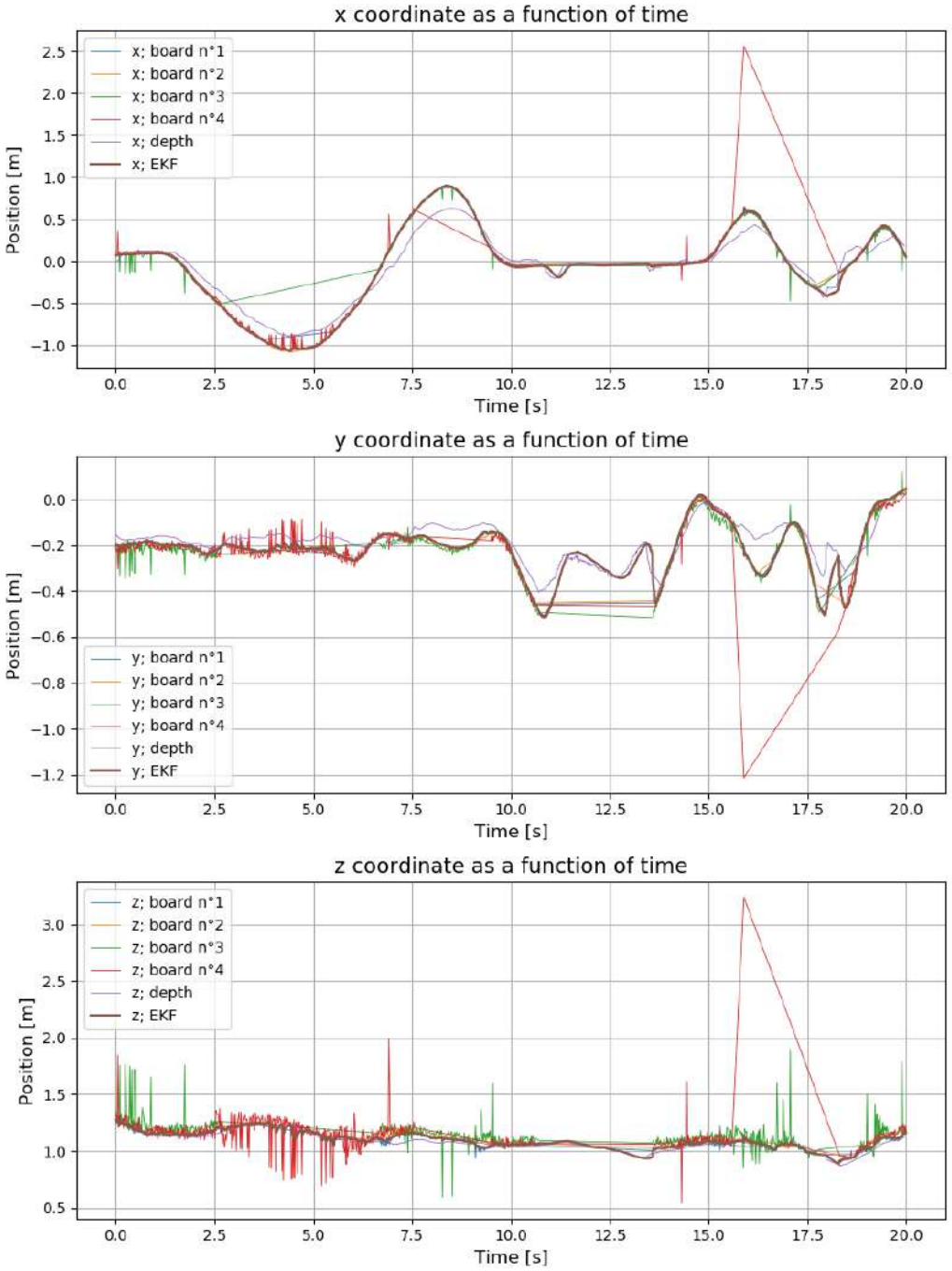


Figure 61: EKF position signals : parameters are described in **Table 23**.

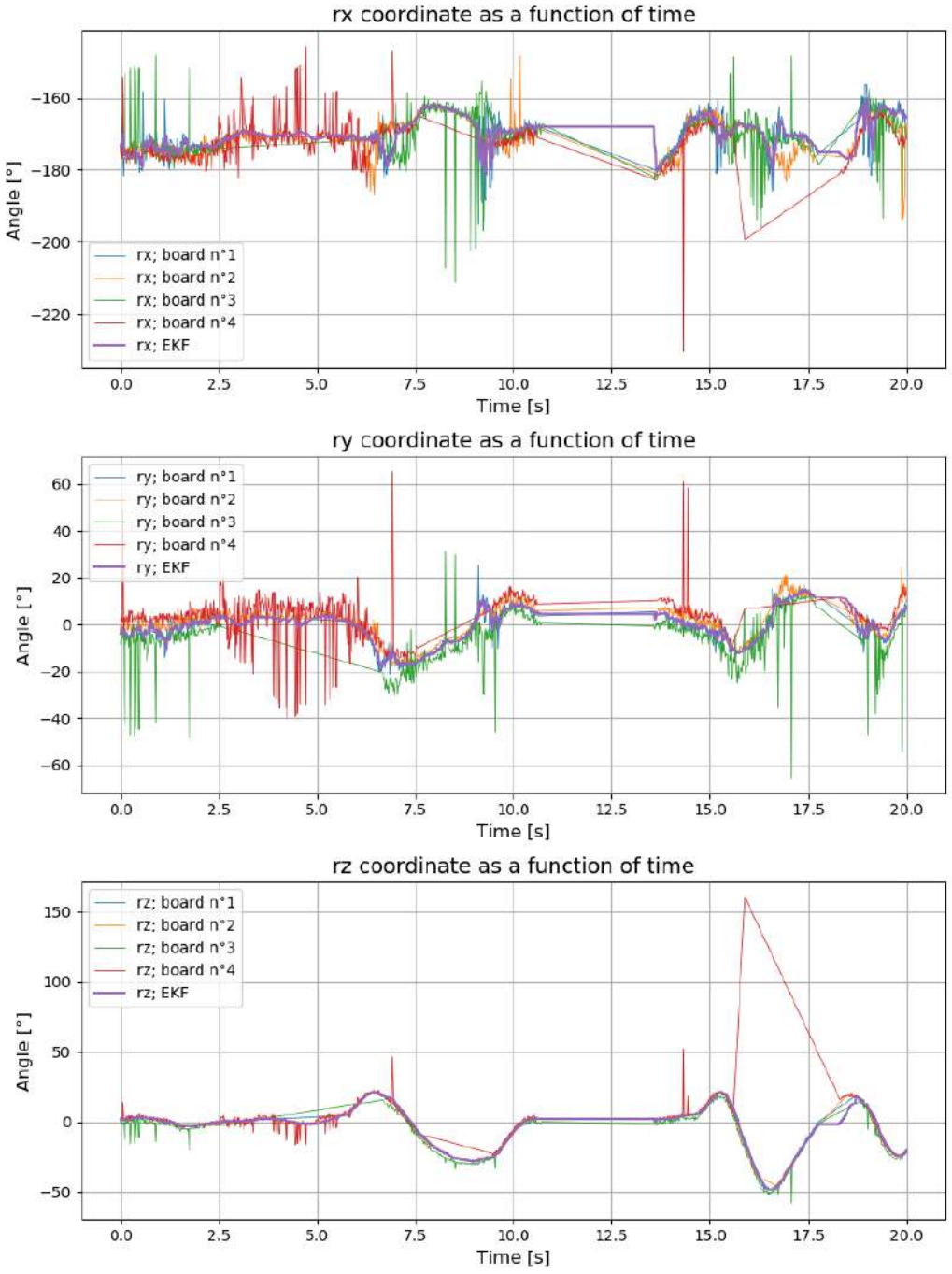


Figure 62: EKF orientation signals : parameters are described in **Table 23**.

These results were achieved using the parameters outlined in **Table 23**.

This tuning might not be optimal, but it provides a solid understanding of our filter's performance. **Table 23** confirms our prior predictions about the noise covariance matrices. Despite some residual noise visible in **Figures 61** and **62**, particularly for roll and pitch, it could be minimized by reducing the corresponding coefficients in the Q matrix. However, excessively relying on the process model can introduce a delay between actual and estimated positions. An example of this delay can be observed in the x coordinate between 10s and 13s, when ArUco markers tracking was lost, and the model shifted to relying solely on depth measurements. As anticipated, this behavior is desired.

Matrix	x	y	z	v_x	v_y	v_z	q_x	q_y	q_z	q_w
Q	0.01	0.01	0.01	0.1	0.1	0.1	0.01	0.01	0.01	0.01
$R_{ArUco,1,2}$	0.1	0.1	0.1	/	/	/	0.1	0.1	0.1	0.1
$R_{ArUco,3,4}$	1	1	1	/	/	/	10	10	10	10
R_{depth}	3	3	3	/	/	/	/	/	/	/

Table 23: Coefficients of Q and R matrices.

In summary, our filter appears to be functioning well. The tracking is smooth, particularly for critical coordinates like 3D positions and yaw rotation. The filter primarily follows the ArUco markers' detections but transitions seamlessly to using depth measurements when tags aren't detected for a certain period. Regarding orientation, due to the lack of input on rotational speeds of both drones, the model assumes constant angles as a result of zero velocities.

It's important to note that these parameters will likely need readjustment for actual flight conditions. One potential avenue to explore is adjusting the R matrices for each measurement source based on the distance between the TD and the SD. For instance, since ArUco detections are less accurate at greater distances from the camera, trusting the depth measurements more could be necessary.

Following this approach, the Extended Kalman Filter emerges as an ideal tool for fusing sensor information. It's evident that GPS is the primary method for detecting the TD when it's not within the camera's FOV, despite its inherent precision limitations. As a result, the coefficient of the R matrix associated with GPS measurements needs to be kept low (indicating trust in the measurement) when the SD is distant from the TD. However, this coefficient should progressively increase as the SD gets closer to the TD. Once the TD enters the camera's FOV, the depth measurement takes precedence over the GPS. The R matrix coefficient associated with the depth measurement, which was previously set high (indicating low trust), should now be decreased. This adjustment reflects the growing reliability of the depth measurement. Ultimately, when the SD is close enough to capture the ArUco markers, these tags should become the primary source of measurement. This strategic transition should be a key aspect of tuning the EKF using real flight data.

To conclude this section on the Extended Kalman Filter, let's briefly examine a video[27] of the tracking to validate our observations.

Figure 63: Screenshots of the EKF results in a live video from different distances.

Figure 63 displays screenshots from a video[27] showcasing the output of our Extended Kalman Filter. In this specific video, the depth measurement was omitted due to the changing z position of the TD. It's evident that the tracking remains consistent and smooth even at distances around 4 meters. This confirms the effectiveness of our EKF under indoor conditions. It's important to acknowledge that outdoor flight scenarios might yield different results, especially considering variations in lighting conditions.

5 Conclusion

The primary objective of this thesis was to design a fast grasping mechanism for the autonomous capture of both aerial and marine drones. This was realized by developing a lightweight and passive gripper affixed to a robotic arm, which could be actuated and managed through ROS. The localization algorithms were constructed to leverage and integrate a diverse array of measurement sources, including GPS, ArUco markers, and depth measurements.

This challenge was initially addressed in Chapter 3, where the mechanical aspects of the project were extensively discussed. The gripper was meticulously designed, taking into consideration the stipulated requirements, enabling a rapid and passive release of the mechanism. The loading system was devised to operate independently from the release mechanism, promoting ease of use. The robotic arm was employed to reposition the catching action away from the propellers, mitigating the adverse effects of downwash turbulence generated by the Shuttle Drone. The electronics were intentionally fashioned to be as simple as possible and seamlessly integratable with the onboard computer.

The evaluation of the Target Drone's localization was conducted in Chapter 4. The objective was to ascertain the relative position of the Target Drone with respect to the Shuttle Drone's body frame, employing various sensors including a camera and depth measurements. While some of the devised algorithms yielded positive outcomes, others did not, yet the collective accuracy of the detection proved satisfactory, largely owing to the application of an Extended Kalman Filter. This filter facilitated the fusion of measurements from diverse sources, contributing to the enhanced precision of the results.

In conclusion, this dissertation will verify the fulfillment of the initial requirements and subsequently assess the constraints inherent in our work. Additionally, potential avenues for future research and development linked to this project will be explored.

5.1 Verification of the Requirements

ReqID	Description	Desired Perf.	Actual Perf.
RSh1	Capture TD	80% Success Rate	To be determined
RSh2	Tracking Accuracy	1cm STD in position 3deg STD in attitude	∼1 cm STD in position ∼5deg STD in attitude
RMe1	Mechanism Mass	$400g \pm 100$	$677g$
RMe2	Repeatability of the Capture Maneuver	10 times ±2	Don't have limits
RMe3	Residual Energy Spent	Minimum	Minimum
RMe4	Tip Waterproof	Fully	Partially

Table 24: Requirements verification.

Table 24 provides a comparison of the desired performance and the actual performance in terms of the requirements for the capture mechanism, for which some comments are provided hereafter. The success rate of the maneuver will be determined in later stages of the CAPTURE project. Measuring the accuracy of the localization algorithms is complex. However, through the application of the Extended Kalman Filter and visual inspection, it can be ascertained that the algorithms are precise enough for successful Target Drone capture. Despite diligent efforts to reduce mechanism mass, the requirement was exceeded by 69%, significantly surpassing the permitted 25%. This predicament arises from the assumption that the mechanism should be positioned away from the Shuttle Drone's propellers. The repeatability of the capture maneuver is virtually unrestricted, limited only by battery levels. The reloading mechanism, decoupled from the closing system, boasts a flawless 100% success rate. The motors used do not consume any power when not in use. Unfortunately, due to time constraints, the prototype isn't fully waterproof, but this issue was evaluated during the thesis, and potential remedies were proposed.

5.2 Limitations of the Thesis and Future Work

Several aspects in the previous dissertation could be enhanced, most of which are influenced by the lack of real flight tests.

Regarding the mechanical and electrical designs, weight reduction to align with the desired requirements may be necessary. Achieving this could involve altering the material and structure of the mechanism, as well as optimizing various design aspects. Concerning the robotic arm, improvements would be contingent on the outcomes of field trials. Current indications suggest that its rigidity may need augmentation, either through adjustments to the diameter of the carbon rods or by shortening their length. If the arm's flexibility is too significant, a method for accurately pinpointing the gripper's tip might become necessary. As expounded in this work, the Target Drone's behavior upon capture is likely to be unpredictable. To address this, the grip on the drone's body might be enhanced through geometric changes

to the gripper. Additionally, active control of the Target Drone post-capture could become necessary.

The accuracy of the localization algorithms will heavily depend on the outcomes of field trials. Presently, these algorithms have only been tested indoors, under constant lighting and controlled conditions. However, these conditions may not necessarily align with real flight tests. The integration of additional measurement sources, including GPS for long-range detection, should also be subjected to testing. Moreover, the depth measurement algorithm needs to be modified, as its current configuration employs a fixed distance for background removal, rather than considering altitude. This thesis discussed various methods of discarding depth measurements. Improving ArUco detection can also be achieved by experimenting with different board configurations on the wings of the Target Drone. It's crucial to emphasize that currently, this remains the sole method for measuring the relative attitude of the Target Drone. The development of alternative algorithms, potentially utilizing GPS or depth measurements, might be necessary. Lastly, proper tuning of the Extended Kalman Filter is essential to ensure optimal performance during outdoor tests.

This thesis was part of the larger project CAPTURE. While initially anticipated to conclude within a 6-month timeframe, we have only achieved a portion of the anticipated outcomes. Further continuation is essential to comprehensively explore this expansive research domain and obtain conclusive results.

References

[1] *CAPTURE Project website*. URL: https://capture.isr.tecnico.ulisboa.pt/ (visited on 08/27/2023).

[2] *Multiplex Kit EasyGlider 4*. pt-pt. URL: https://www.svmodelismo.net/pt/planadores/2134-multiplex-kit-easyglider-4.html (visited on 08/28/2023).

[3] *M690B T-DRONES T-MOTOR*. URL: https://store.tmotor.com/goods-831-M690B.html (visited on 08/28/2023).

[4] Evan Ackerman and Michael Koziol. "The blood is here: Zipline's medical delivery drones are changing the game in Rwanda". In: *IEEE Spectrum* 56.5 (May 2019). Conference Name: IEEE Spectrum, pp. 24–31. ISSN: 1939-9340. DOI: 10.1109/MSPEC.2019.8701196.

[5] Julian Rothe, Michael Strohmeier, and Sergio Montenegro. "A concept for catching drones with a net carried by cooperative UAVs". In: *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. ISSN: 2475-8426. Sept. 2019, pp. 126–132. DOI: 10.1109/SSRR.2019.8848973.

[6] William Stewart et al. "How to Swoop and Grasp Like a Bird With a Passive Claw for a High-Speed Grasping". en. In: *IEEE/ASME Transactions on Mechatronics* 27.5 (Oct. 2022), pp. 3527–3535. ISSN: 1083-4435, 1941-014X. DOI: 10.1109/TMECH.2022.3143095. URL: https://ieeexplore.ieee.org/document/9697910/ (visited on 03/07/2023).

[7] Andrew McLaren et al. "A Passive Closing, Tendon Driven, Adaptive Robot Hand for Ultra-Fast, Aerial Grasping and Perching". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866. Nov. 2019, pp. 5602–5607. DOI: 10.1109/IROS40897.2019.8968076.

[8] Bhivraj Suthar and Seul Jung. "Design of a Drone-Hawk Gripper for Nest Removal from Electric Tower". In: *2021 21st International Conference on Control, Automation and Systems (ICCAS)*. ISSN: 2642-3901. Oct. 2021, pp. 1952–1954. DOI: 10.23919/ICCAS52745.2021.9650012.

[9] W. R. T. Roderick, M. R. Cutkosky, and D. Lentink. "Bird-inspired dynamic grasping and perching in arboreal environments". en. In: *Science Robotics* 6.61 (Dec. 2021), eabj7562. ISSN: 2470-9476. DOI: 10.1126/scirobotics.abj7562. URL: https://www.science.org/doi/10.1126/scirobotics.abj7562 (visited on 08/29/2023).

[10] HaoTse Hsiao et al. "A Mechanically Intelligent and Passive Gripper for Aerial Perching and Grasping". In: *IEEE/ASME Transactions on Mechatronics* 27.6 (Dec. 2022). Conference Name: IEEE/ASME Transactions on Mechatronics, pp. 5243–5253. ISSN: 1941-014X. DOI: 10.1109/TMECH.2022.3175649.

[11] Justin Thomas et al. "Toward image based visual servoing for aerial grasping and perching". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. ISSN: 1050-4729. May 2014, pp. 2113–2118. DOI: 10.1109/ICRA.2014.6907149.

[12] Tong Li et al. "An Efficient Network for Target-oriented Robot Grasping Pose Generation in Clutter". In: *2022 IEEE 17th Conference on Industrial Electronics and Applications (ICIEA)*. ISSN: 2158-2297. Dec. 2022, pp. 967–972. DOI: 10.1109/ICIEA54703.2022.10005947.

[13] dafonso. *ISR website*. en-US. URL: https://welcome.isr.tecnico.ulisboa.pt/ (visited on 08/29/2023).

[14] *REPLACE Project website*. URL: https://replace.isr.tecnico.ulisboa.pt/ (visited on 08/28/2023).

[15] Volodymyr Kharchenko and Iurii Chyrka. "Detection of Airplanes on the Ground Using YOLO Neural Network". en. In: *2018 IEEE 17th International Conference on Mathematical Methods in Electromagnetic Theory (MMET)*. Kiev: IEEE, July 2018, pp. 294–297. ISBN: 978-1-5386-5438-5. DOI: 10.1109/MMET.2018.8460392. URL: https://ieeexplore.ieee.org/document/8460392/ (visited on 06/09/2023).

[16] Manuel García et al. "Autonomous drone with ability to track and capture an aerial target". In: *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. ISSN: 2575-7296. Sept. 2020, pp. 32–40. DOI: 10.1109/ICUAS48674.2020.9213883.

[17] Bård Nagy Stovner et al. "Attitude estimation by multiplicative exogenous Kalman filter". en. In: *Automatica* 95 (Sept. 2018), pp. 347–355. ISSN: 00051098. DOI: 10.1016/j.automatica.2018.05.038. URL: https://linkinghub.elsevier.com/retrieve/pii/S0005109818302802 (visited on 07/30/2023).

[18] *Spring Reference*. fr. URL: https://www.ressorts-sodemann.fr/41140 (visited on 03/23/2023).

[19] *Linear Pusher 30mm*. URL: https://eu.mouser.com/ProductDetail/DFRobot/FIT0804?qs=pBJMDPsKWf0ip5B%252BVQy3og%3D%3D (visited on 08/28/2023).

[20] Yaozong Zhu et al. "CFD simulation and measurement of the downwash airflow of a quadrotor plant protection UAV during operation". en. In: *Computers and Electronics in Agriculture* 201 (Oct. 2022), p. 107286. ISSN: 0168-1699. DOI: 10.1016/j.compag.2022.107286. URL: https://www.sciencedirect.com/science/article/pii/S0168169922005981 (visited on 06/21/2023).

[21] *Linear Pusher 50mm*. URL: https://eu.mouser.com/ProductDetail/DFRobot/FIT0805?qs=pBJMDPsKWf1T7nMWxYK%2FVA%3D%3D (visited on 08/28/2023).

[22] *Servo motor Reference*. URL: https://www.dsservo.com/en/download.asp (visited on 05/30/2023).

[23] *IntelRealSense realsense-ros GitHub*. original-date: 2016-02-23T23:52:58Z. Aug. 2023. URL: https://github.com/IntelRealSense/realsense-ros (visited on 08/28/2023).

[24] Baej Sowa. *ArUco Ros detection*. URL: https://github.com/bjsowa/aruco_ros (visited on 08/28/2023).

[25] Marcelo Jacinto. *Marcelo Jacinto GitHub*. en. URL: https://github.com/MarceloJacinto (visited on 08/28/2023).

[26] *Extended Kalman filter*. en. Page Version ID: 1170100495. Aug. 2023. URL: https://en.wikipedia.org/w/index.php?title=Extended_Kalman_filter&oldid=1170100495 (visited on 08/28/2023).

[27] Arthur Dietrich. *Attached Videos*. Google Drive. URL: https://drive.google.com/drive/folders/1RCl7Mj0BbXZndmDnfom3_DSUAc0-K3HC?usp=sharing.

[28] u-blox. *NEO-M8 u-blox M8 concurrent GNSS modules*. 2022. URL: https://www.u-blox.com/en/product/neo-m8-series.

[29] Bruno Guerreiro et al. *D2.2.1 Requirements and possible designs for shuttle and target drones*. en. Tech. rep. 2021.

[30] Texas Instrument. *PTN78000w Datasheet*. Tech. rep. 2004. URL: https://www.ti.com/lit/ds/symlink/ptn78000w.pdf?ts=1693398479481&ref_url=https%253A%252F%252Fwww.google.com%252F.

[31] Francisco Azevedo. "Automatic grasping system for multi-drone parcel delivery". MSc Thesis in Mechanical Engineering. Instituto Superior Técnico, 2019.

[32] *Interface L298N DC Motor Driver Module with Arduino*. en-GB. Nov. 2018. URL: https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/ (visited on 06/28/2023).

[33] *Intrinsic Camera Parameters - D435*. en. URL: https://github.com/IntelRealSense/realsense-ros/issues/709 (visited on 04/11/2023).

[34] Jeff Faudi. *Aircraft Detection with YOLOv5*. en. URL: https://kaggle.com/code/jeffaudi/aircraft-detection-with-yolov5 (visited on 07/18/2023).

[35] Marcelo Jacinto. *Pegasus Simulator*. URL: https://pegasussimulator.github.io/PegasusSimulator/source/tutorials/create_custom_backend.html (visited on 03/06/2023).

[36] *ChatGPT*. URL: https://chat.openai.com (visited on 08/27/2023).

[37] *Compare Intel RealSense Depth Cameras (Tech specs and Review)*. en-US. URL: https://www.intelrealsense.com/compare-depth-cameras/ (visited on 08/20/2023).

[38] *rosserial - ROS Wiki*. URL: https://wiki.ros.org/rosserial (visited on 08/20/2023).

[39] *Calculate X, Y, Z Real World Coordinates from Image Coordinates using OpenCV*. en-US. Apr. 2019. URL: https://www.fdxlabs.com/calculate-x-y-z-real-world-coordinates-from-a-single-camera-using-opencv/ (visited on 08/05/2023).

[40] *Intelő RealSense Depth Camera D455 - Product Specifications*. en. URL: https://www.intel.com/content/www/us/en/products/sku/205847/intel-realsense-depth-camera-d455/specifications.html (visited on 08/05/2023).

[41] *M690B Wiki*. URL: https://hardtekpt.github.io/M690B-Wiki/ (visited on 07/29/2023).

[42] Dejan. *L298N Motor Driver - Arduino Interface, How It Works, Codes, Schematics*. en-US. Running Time: 599. Aug. 2017. URL: https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/ (visited on 06/28/2023).

[43] Sarah Whittaker. *Origami-Inspired Foldable Robot Arm Developed for Drones*. en-US. Mar. 2018. URL: https://dronebelow.com/2018/03/15/origami-inspired-foldable-robot-arm-developed-for-drones/ (visited on 05/17/2023).

[44] Asadullah Dal. *Basic-Augmented-reality-course-opencv*. original-date: 2022-01-27T08:26:43Z. Feb. 2023. URL: https://github.com/Asadullah-Dal17/Basic-Augmented-reality-course-opencv (visited on 03/20/2023).

[45] *Motor constants*. en. Page Version ID: 1137142274. Feb. 2023. URL: https://en.wikipedia.org/w/index.php?title=Motor_constants&oldid=1137142274 (visited on 03/20/2023).

[46] Manohar B. Srikanth, Hari Vasudevan, and Manivannan Muniyandi. "DC Motor Damping: A Strategy to Increase Passive Stiffness of Haptic Devices". en. In: *Haptics: Per-*

ception, Devices and Scenarios. Ed. by David Hutchison et al. Vol. 5024. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 53–62. ISBN: 978-3-540-69056-6 978-3-540-69057-3. DOI: 10.1007/978-3-540-69057-3_6. URL: https://link.springer.com/10.1007/978-3-540-69057-3_6 (visited on 03/16/2023).

[47] *Collision parfaitement inélastique*. fr. Page Version ID: 155150018. Dec. 2018. URL: https://fr.wikipedia.org/w/index.php?title=Collision_parfaitement_in%C3%A9lastique&oldid=155150018 (visited on 03/14/2023).

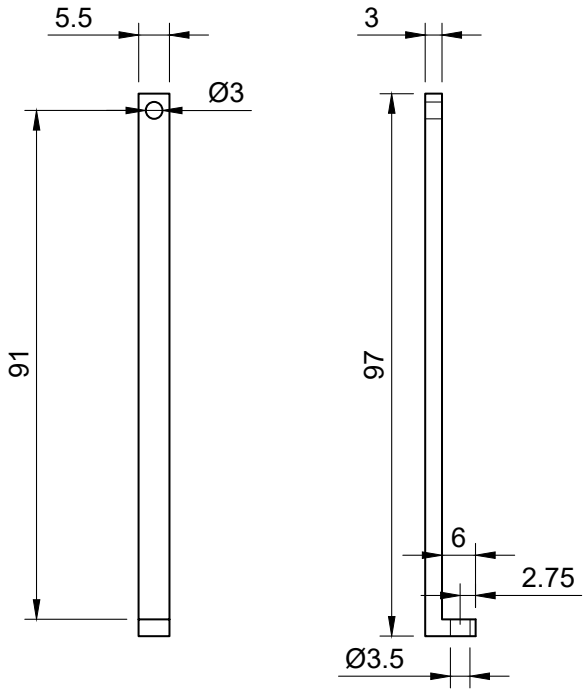
[48] Duncan W. Haldane et al. "Robotic vertical jumping agility via series-elastic power modulation". en. In: *Science Robotics* 1.1 (Dec. 2016), eaag2048. ISSN: 2470-9476. DOI: 10.1126/scirobotics.aag2048. URL: https://www.science.org/doi/10.1126/scirobotics.aag2048 (visited on 03/07/2023).

[49] Daniel Mellinger and Vijay Kumar. "Minimum snap trajectory generation and control for quadrotors". In: *2011 IEEE International Conference on Robotics and Automation*. ISSN: 1050-4729. May 2011, pp. 2520–2525. DOI: 10.1109/ICRA.2011.5980409.

[50] Tony G. Chen et al. "Aerial Grasping and the Velocity Sufficiency Region". In: *IEEE Robotics and Automation Letters* 7.4 (Oct. 2022). Conference Name: IEEE Robotics and Automation Letters, pp. 10009–10016. ISSN: 2377-3766. DOI: 10.1109/LRA.2022.3192652.

Appendices

A 2D Drawing



Dept.	Technical reference	Created by Arthur Dietrich 8/15/2023	Approved by
		Document type	Document status
		Title plate_holder	DWG No.
			Rev. / Date of issue / Sheet 1/1