POLITECNICO DI TORINO

MASTER's Degree in COMPUTER ENGINEERING

MASTER's Degree Thesis

Drones in Warehouses: Advancements in Autonomous Inventory

A Comparative Analysis of Vision-Based Techniques and Fiducial Marker Families for Indoor Drone Navigation and Relocation

Supervisors

MARCELLO CHIABERGE

SIMONE GODIO

Candidate

LUCA GENOVESE

DECEMBER 2023

Abstract

Autonomous navigation is one of the most interesting and complicated challenges of recent years. It is important to consider and evaluate the technical and application complexity involved in the autonomous flight of a drone within an indoor environment, where fixed and moving obstacles are encountered. Nowadays, there is a continuous attempt to automate and optimize any kind of process, to support humans work, and where possible, make machines do the most repetitive, alienating, time-consuming, error pruning and dangerous jobs. In this way humans can focus on activities that enhance their intellectual value and business can grow faster and faster, by investing more and more in autonomous technology solutions.

Thanks to today's cutting-edge technologies, drones can therefore be used to facilitate inventory and logistics operations in traditional warehouses, to carry out these operations in a safer, more efficient, and completely autonomous way.

The state of the art regarding these issues is studied, and the main techniques and technologies that can be used to locate and navigate autonomously within an indoor environment are briefly explained. Solutions in which a standalone drone is used will be compared with others where the drone is supported by an AMR. After a study of the main indoor localization techniques such as UWB, wheel odometry, Lidar odometry etc; the main vision-based techniques are discussed more in-depth: VIO and fiducial marker-based relocalization, which often work in a complementary way with each other.

Finally, the results of an extensive study of 2 families of fiducial markers are presented: Aruco vs April-Tag. The main differences in accuracy and performance using different metrics. This is done by comparing measurements obtained from 4 different cameras: two with a standard aperture, one wide-angle camera, and a fisheye camera. The AprilTags camera demonstrated better performance than ArUco markers, with a broader working range and higher pose stream frequency. The D435i camera was found most suitable for reliable and precise relocalization, maintaining high data stream frequency. Its balance between a webcam with a limited field of view and a camera with a wide FOV makes it ideal for autonomous drone navigation. The D435i camera is lightweight, compact, energy-efficient, and equipped with a CPU for obstacle avoidance. Relocalization based on fiducial markers can be implemented if supported by proper sensor fusion algorithms and filters.

Summary

This master's thesis explores the topic of autonomous inventory management conducted by drones. It delves into its various aspects: social, technical/technological, and economic impact on businesses that choose to adopt these technologies.

Chapter 1: In this chapter, a general introduction to inventory management is provided, explaining the traditional procedure for inventory operations conducted by businesses, while analyzing the associated issues, challenges, and limitations. Furthermore, an economic perspective is discussed, emphasizing the critical importance of accurate inventory management for businesses and the substantial financial losses resulting from minor daily inventory errors: there are current news examples of major companies incurring million-dollar losses for these reasons. Following this, an overview of drones topic globally, in Europe, and particularly in Italy is given, analyzing the main sectors where these devices are utilized, the types of solutions offered, and the scale of investments related to this topic. The analysis suggests these are exciting years for the drone industry, with promising applications emerging in various sectors. It is reported that "A total of €980 million has been allocated to the development or use of drones for cutting-edge applications", "The drone services market size is expected to grow to $63.6 billion by 2025" and finally "Start-ups received capital of $7 billion in recent years, of which $17 million was allocated to logistics". The use of drones in the next 5-10 years seems particularly innovative and promising, namely in logistics, where the main use case would be the autonomous drone inventory. Then are present the key benefits in terms of efficiency, autonomy, quality, safety, and costs, as well as the significant challenges posed by these technologies, such as indoor drone localization in a warehouse. These challenges will be further examined throughout the thesis, with possible solutions presented and proposed.

Chapter 2: Following an introduction to automation in logistics, including other use cases, a detailed examination of inventory typology is conducted, providing the reader with a deeper understanding of this operational process. After a brief explanation of the levels of automation, are presented some production-ready solutions for autonomous inventory already available on the market today. First, there are very basic semi-autonomous solutions in which the drone is manually

piloted to capture photos at each location, progressing gradually to solutions with minimal autonomous logic in which drones can navigate autonomously using QR codes. However, these solutions may require operator intervention due to limited sensor capabilities. The chapter then explores more advanced autonomous solutions featuring drones equipped with the necessary sensors for safe independent and autonomous navigation. Stand-alone drone solutions are scalable, allowing a swarm of drones to work simultaneously in different parts of a warehouse. Other solutions include drones being supported by ground-based Autonomous Mobile Robots (AMRs) for improved localization. AMRs excel at localization compared to drones, making the drone follow the AMR's commands and rely on the ground robot for localization. There are variations of these AMR-assisted solutions, including those providing power to the drone (through a cable), enabling longer system autonomy. Some solutions also incorporate telescopic poles mounted on AMRs for capturing images at higher locations. The pros and cons of each of these solutions are analyzed in detail.

Chapter 3: This chapter provides an overview of indoor localization techniques, explaining the concept of SLAM (Simultaneous Localization and Mapping). It delves into camera-based SLAM and Lidar-based SLAM in more detail. Within camera-based SLAM, Visual Odometry (VO) and Visual Inertial Odometry (VIO) are discussed. These techniques use environmental features to enable localization and navigation by calculating the camera's movement in space. Wheel odometry, which estimates the pose of a ground-based robot using wheel encoders, is also explained, albeit this technique is not applicable to drones. The chapter highlights the main challenge with these odometric techniques: drift error accumulates over time during navigation, but it can be reset using techniques like fiducial marker relocation. Additionally, alternative localization techniques based on pre-installed antenna constellations are examined, such as Ultra Wide Band (UWB). UWB triangulates the signal from a receiver antenna on the object to provide a real-time, highly precise (cm-accuracy) position estimation. However, UWB systems face complications during antenna installation and are susceptible to interference in environments like warehouses, particularly due to metal shelving that led to complications. With fiducial markers is not possible real-time localization, but this technique is important for relocalizing objects based on specific markers identified during navigation, it is used in conjunction with odometric techniques (like VIO) to reset drift errors. Finally, it offered an overview of various Kalman filter models and their specific characteristics. Kalman filters enable sensor fusion, combining data streams from different sensors or cameras while assigning varying weights to each sensor based on data accuracy and reliability. In essence, if multiple systems provide real-time position data for a robot, such as VIO, wheel odometry, and UWB, the Kalman filter calculates a weighted average to produce a single (fused) position estimate that takes all three data sources into account. Then if one sensor, such as VIO, fails to work properly, the Kalman filter “understands” this failure and excludes its data from the final fused position estimate.

Chapter 4: This chapter, a comprehensive study of the techniques introduced in the previous chapter, is structured in two phases. The first phase involves implementing ROS (Robot Operating System) nodes to test and measure position data obtained from VIO (with the T265 camera) and a basic webcam. During this phase, a node is developed to estimate the position based on identified markers (ArUco) in the image frame. The idea is to combine these two position data sets from VIO and marker-based relocalization in a straightforward manner, without applying additional logic or specific filters (e.g., EKF) to enhance the final position. Based on initial results that showed imprecise position estimation using fiducial markers, a more in-depth study of this technique is conducted in the second phase. In the second phase, measurements are taken with two families of fiducial markers (AprilTag and ArUco) and four different types of cameras (Logitech C270 webcam, RasPi Cam IMX219-160 IR, Intel Realsense T265, and Intel Realsense D 435i). The aim is to understand how the position estimate varies with changes in distance, camera type, marker type, and marker size (two sets of measurements with markers of 13.8 cm and 28 cm sides). The analysis is performed separately for the X, Y, and Z axes, with the results analyzed to draw conclusions in the final chapter.

Chapter 5: The final chapter of the master thesis presents a comprehensive analysis of measurement sets for the X, Y, and Z axes, comparing the performance of AprilTags and ArUco markers across various cameras. The D435i camera emerges as the most suitable choice for reliable and precise relocalization, offering a balanced performance in terms of stability, accuracy, and pose stream frequency. The usage of AprilTags with the D435i camera is identified as a promising approach for implementing relocalization based on fiducial markers for drone navigation in indoor environments. It explains the importance of marker design, specifically addressing marker size considerations for optimal performance. Furthermore, it is proposed the implementation of a dedicated logic to filter out poses near specific thresholds, addressing challenges such as Z-flipping and reduced accuracy near range margins or close to the perpendicular axis of the maker. The conclusion also outlines future research directions, including a multi-variable comparative study, experiments with different markers and infrared cameras, and exploration of event-based cameras. Additionally, suggestions for improving markers themselves are presented, ranging from enhancements to existing marker families to the development of entirely new marker families based on color information. These potential advancements aim to further enhance the accuracy and reliability of pose estimation in relocalization applications.

Acknowledgements

Grazie a mamma e papà,

oggi, dopo molti anni di sacrifici, sconfitte, vittorie, sfide, difficoltà, obiettivi raggiunti, fallimenti e successi, si conclude un capitolo molto importante della mia vita. Ho sputato sangue, versato lacrime, avuto periodi difficili, ma papà e mamma erano sempre lì al mio fianco, pronti a supportarmi e sopportarmi, a fare il tifo per me, e a darmi la forza di non mollare e non arrendermi davanti alle difficoltà.

In questo percorso di laurea magistrale ho avuto la fortuna di fare le esperienze più disparate e di uscire dalla tanto famosa comfort zone. L'"Erasmus" (mobilità extra UE per i più puntigliosi) in Brasile è stata una delle esperienze più belle della mia vita, l'ha cambiata la mia vita, mi ha insegnato a vivere la vita con occhi diversi... come dico sempre, prima del Brasile ero un "ingegnere grigio", ora sono un "ingegnere colorato", quale sia la differenza non è facile da spiegare, ma questo colore ce lo si sente dentro.

Mamma e papà, anche se non lo ammetteranno mai, non erano molto felici quel 27 febbraio 2020 (sì, giusto qualche giorno dopo lo scoppio del Covid in Italia), perché quel giorno mi sarei imbarcato su un aereo con destinazione San Paolo... Beh quel giorno o meglio quella notte, alle 4 di mattina, tra gli abbracci e le lacrime di felicità mista a commozione, loro erano lì a supportarmi.

Voglio ringraziare i meninos e le meninas che ho conosciuto in Brasile, con i quali abbiamo superato la quarantena tra churrascos in repi, galoppate a cavallo sul bagnasciuga al tramonto, bagni in piscina e corsette infinite in un campus deserto, in cui regnava un silenzio assordante. Grazie ai ragazzi della Picareta, che hanno condiviso con me i mesi più difficili del covid, nei quali con una pazienza enorme mi hanno insegnato il portoghese giorno dopo giorno, valeu manos! Gente boa!

Voglio citare poi una frase di uno di questi ragazzi, Jonas, che con la sua energia, positività, sorriso e spensieratezza ha condiviso con me la maggior parte della quarantena, solo io e lui, tra lezioni di chitarra fallimentari, preparazioni di pop-corn a base di dado Knorr e kg di abacate ingeriti per 1 settimana, tutti i giorni. Era la persona giusta che con il suo carattere mi ha aiutato a smussare i miei spigoli, e a fare uscire l'ingegnere colorato che era dentro di me, e che è dentro ognuno di noi!

O Gentil (nessuno sa perchè fosse soprannominato così, ma tutti concordavano che fosse il soprannome più azzeccato) una sera disse: "Cada pessoa que você conhecer deixará um pedacinho dela dentro de você e, aop mesmo tempo, você deixará um pedacinho de você em todas pessoas que você vai encontrar no seu caminho", lui la diceva meglio...

Anche se tutto ciò può spaventare e sembra troppo difficile, irraggiungibile o troppo rischioso, tutti dovremmo fare almeno una volta nella vita un'esperienza studentesca e/o lavorativa di questo tipo. Sono esperienze che ci forzano a spingerci oltre i nostri limiti, e, in un modo o in un altro, faranno crescere. Ora siamo giovani, oggi non abbiamo vincoli.

Ora tutti noi possiamo darci, dobbiamo darci la possibilità di fallire, perché tra qualche anno sarà troppo tardi.

Voglio ringraziare anche tutti i compagni di studio di questi anni: con alcuni ci conosciamo dalle medie, con altri dal liceo, altri li ho conosciuti in università per pochi mesi, mentre con altri ancora abbiamo condiviso quasi tutto il percorso universitario... con ognuno di voi, chi più chi meno abbiamo sofferto tante agonie, patito ansie, passato notti insonni pre-esame... ma tutto ciò era contornato da qualche piccola gioia ogni tanto (ma mai troppe da montarci la testa). Un grazie anche ai colleghi che periodicamente, ogni 2/3 mesi, mi chiedevano a che punto fossi con la tesi, mettendomi ansia ma spronandomi allo stesso tempo ahaha!

Come diceva o Gentil, ognuno di voi ha lasciato un pezzetto del suo essere dentro di me, che fa di me ciò che sono oggi.

Voglio ringraziare gli zii, Rita, Simone, il piccolo Davide e Dario che hanno sempre fatto il tifo per me standomi vicino quando ero lontano da casa e durante questo ultimo particolare periodo.

Voglio poi ringraziare lo sport, che in questi anni mi ha sempre dato tanto, è stato spesso un rifugio sicuro, ma anche una valvola di sfogo nei momenti di tensione. Lo sport mi ha permesso di conoscere moltissimi amici che voglio ringraziare tra cui Sara, "Andromeda" e tutti gli altri compagni di pallonate, sciate, camminate e altre disparate attività...

Ringrazio poi il Professor Chiaberge e il Dottor Godio che hanno deciso di accompagnarmi per il viaggio finale di questa avventura, per la loro pazienza e il loro supporto accademico, ma soprattutto umano.

Voglio concludere i ringraziamenti condividendo una cosa che purtroppo è successa qualche mese fa, non voglio fare il moralista o il "saggio guru" come direbbe una mia amica, che ringrazio nuovamente in modo particolare e sentito, per essermi stata vicino in questi anni, ma soprattutto in questi mesi.

La vita è una ruota che gira, ci dà tantissimo, ma sa anche metterci duramente alla prova. Ad aprile stavo ultimando questa tesi, ero in un periodo positivo della mia vita, ero riuscito a realizzare un sogno a cui stavo pensando da tempo, ma la vita è beffarda, e solo 3 giorni dopo la realizzazione di quel sogno, quello stesso sogno si è trasformato in un bruttissimo incubo. Un membro della mia famiglia, che è sempre stato trainante, forte, energico, positivo e sorridente si è trovato, per una fatalità che solo il destino può spiegare, ad essere il protagonista di questo incubo.

Io voglio dedicare il più grande grazie di questa lunga serie di ringraziamenti a te PAPA', per essere così forte, perché nonostante ciò che è successo hai continuato a sorridere, ad essere positivo, energico e combattivo come prima, anzi, più di prima! Tu hai detto no, non ci sto, io vado avanti lo stesso. Sono fortunato ad avere un papà gladiatore: questo è il soprannome che uso quando parlo di te. Quello che mi hai insegnato tu in questi mesi, con la tua forza d'animo, la tua volontà di sorridere alla vita nonostante tutto, e il tuo coraggio di superare anche questa prova che il destino ci ha presentato, beh purtroppo è una lezione che non si trova in nessun libro né corso universitario. Quello che mi porto dietro è la consapevolezza di dover e poter affrontare qualsiasi conto che la vita ci presenti con il cuore e con la testa, come hai fatto e stai facendo tu in questi mesi. La lucidità di rialzarsi in piedi dopo qualsiasi caduta e andare avanti nonostante tutto.

Giorno per giorno, passo dopo passo, gradino dopo gradino... fino ad arrivare in vetta e urlare al mondo che ce l'hai fatta, che ce l'abbiamo fatta!

La vita è oggi e va vissuta, domani è troppo tardi per farsela raccontare.

Grazie papà e mamma.

Table of Contents

List of Figures

Acronyms

VIO
Visual Inertial Odometry

VO
Visual Odometry

OF
Optical Flow

AGV
Automated Guided Vehicles

SLAM
Simultaneous Localization and Mapping

ROS
Robot Operating System

AMR
Autonomous Mobile Robot

ASRS
Automated Storage and Retrieval System

EKF
Extended Kalman Filter

LOS
Line Of Sight

WMS
: Warehouse Management System

JARUS
: Joint Authorities for Rulemaking on Unmanned Systems

HOTL
: Human on the Loop

HOTL
: Human in the Loop

ODD
: Operational Design Domain

ENAC
: Ente Nazionale per l'Aviazione Civile

LIDAR
: Light Detection and Ranging

PnP
: Perspective-n-Point

FOV
: Field Of View

Chapter 1

Drone and Logistics

1.1 Introduction

In recent years, more and more efforts are being made to automate as much as possible all processes that are repetitive or in some way wearisome and dangerous for human beings. Thinking about the whole logistics process within a warehouse, we want to focus on one activity in particular that is frequently performed within all warehouses: stocktaking. This activity consists of scanning in a very repetitive way, all the locations within a warehouse: the operator normally equipped with a barcode or QR-code reader must go in front of every specific location and scan the codes he finds attached to the pallet or boxes.

The main objective of the inventory is to ensure that the logical and physical match, meaning that the information on the WMS (Warehouse Management System) describing the quantity and the location of products inside the warehouse are correct. In case inconsistencies are found at the end of the inventory, the WMS automatically aligns "the logical to the physical" i.e., corrects and updates the new and actual physical situation with the info on the WMS.

1.1.1 Importance of inventory for companies and businesses

The inventory operation in itself is very simple to perform: once the warehouse manager schedules an inventory using a WMS, tasks/missions are automatically generated for an operator to scan a certain set of locations. When he starts doing the inventory, the operator already knows which locations the inventory he is performing will insist on: it can be all the locations in a small warehouse, all the locations in a specific area, or only a set of locations that meet certain criteria. The user will just need to go to the locations on the list (paper or digital) that his mission specifies, scan the item's code, and enter this information into the WMS. The operator will be guided from one location to another if the system is not paper but digital (app or other), and once he reaches the location, the system will aid him through the steps he needs to conduct step by step.

Code scanning is the "basic" inventory, but it is also possible that during this operation the operator will be asked to enter the quantities of the items, some additional details like the lot or expiration date of the goods or other info, and finally can be asked to take photos and thus perform quality control of the goods contextually with the inventory, optimizing in the same time other processes.

The verb “inventory” refers to the act of counting or listing items, but as an accounting term, inventory is a current asset and refers to all stock in the various production stages. This second meaning of the term inventory describes the variety of goods, products, or materials a company has on hand for consumption, production, or distribution. The stock comprises all sellable goods while inventory includes all of the items needed to produce, stock, and distribute the goods, as well. It functions as a sizable asset and typically consists of components, finished goods, raw materials, works-in-progress, maintenance, repair and operations (MRO) goods. Maintaining the ideal balance between supply and demand, ensuring that sufficient stock levels are maintained to meet customer orders and production requirements, and avoiding overstock that ties up capital and results in additional costs are all part of effective inventory management.

In general, inventory allows the business to replenish low-level stock, determine which is the correct number of items in the event of differences, and eventually open a new count. In these situations, it is crucial to investigate the potential causes of the discrepancies, examine the stocktake’s results, and identify areas that could be improved.

Inventory is the most important asset many organizations hold, representing as much as half of the company’s expenses, or even half of the total capital investment. In addition, according to Science Direct, the past two decades have seen an increase in inventory management research interest. As shown in Figure 1.1, the publication of articles on inventory management has seen an increase of over 525 percent, with the number of published articles increasing from 2544 in 1998 to 13381 in 2020 [1].

Consider Ted Baker, which estimated in January 2019 that it had lost £25 million due to overstated stock [2]; after a review by a third-party firm, it later emerged that the initial estimates of a stock miscalculation had more than doubled, stating it had overestimated the numbers by at least £58 million. “Ironically, the inventory mountain at Ted Baker is referred to as a £58m hole,” said Ian Smith, finance director, and general manager at automated accounts payable and document management, “I would suggest it is not such a surprise that they have an inventory management issue. Due to Ted Baker’s misrepresentation of stock, the company’s profitability would have been overestimated, which would have had an impact on the share price on the London Stock Exchange. This example illustrates the importance of inventory and how small errors in stock valuation can have far

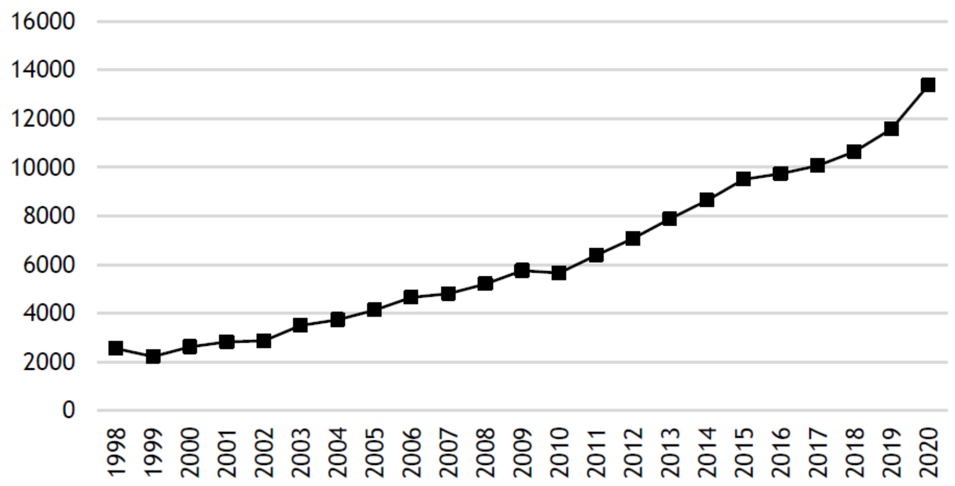


Figure 1.1: Science Direct publications on inventory management, 1998—2020

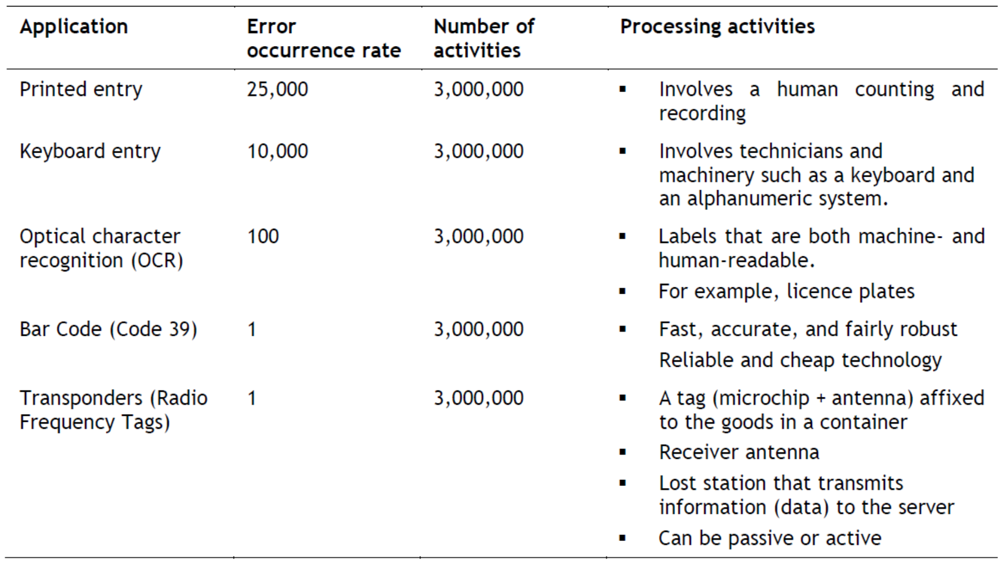
more serious consequences.

1.1.2 Problems and dangers of inventory

These locations, however, in very large warehouses can be as much as tens of meters high above the ground, and therefore, the operator must use scissor-lifters or other equipment that allows him to reach the highest shelves and thus scan the locations at height. The use of these scissor-elevators, however, has inherent problems: it is an inefficient task in terms of movement because in any case, the time required to lift the machine is not negligible, the operator who finds himself performing this task is still exposed to an intrinsic risk due to the fact that he has to take himself to heights (up to even 20 m) to perform the scan, and finally, it is a very repetitive and error-prone task that in the long term leads to alienation and thus committing distracting errors and thus reducing the effectiveness and accuracy of the task. Figure 1.2 shows the error occurrence rate in processing activities using inventory management applications.

The results show that the less daily human handling of stocks there is, the fewer the errors [1].

The operations mentioned above are extremely risky; in fact, several accidents (some more serious than others) take place in warehouses every year. As a result, it is a good idea for businesses to push for the automation of these operations using robots and autonomous systems in a completely safe manner. For example in Sidney in May 2016, a man lost his right leg following an accident at a warehouse in Sydney in May 2016 [3]. The victim said he had gone out to 'look for some material'



Application	Error occurrence rate	Number of activities	Processing activities
Printed entry	25,000	3,000,000	▪ Involves a human counting and recording
Keyboard entry	10,000	3,000,000	▪ Involves technicians and machinery such as a keyboard and an alphanumeric system.
Optical character recognition (OCR)	100	3,000,000	▪ Labels that are both machine- and human-readable. ▪ For example, licence plates
Bar Code (Code 39)	1	3,000,000	▪ Fast, accurate, and fairly robust Reliable and cheap technology
Transponders (Radio Frequency Tags)	1	3,000,000	▪ A tag (microchip + antenna) affixed to the goods in a container ▪ Receiver antenna ▪ Lost station that transmits information (data) to the server ▪ Can be passive or active

Figure 1.2: Rates of error occurrence in activities

before a 'split second lack of concentration' cost him a limb. His testimony was: 'During that process, I took a step back to look upwards and the guy on the big side-loader forklift collided with me and dragged my right foot underneath the forklift'. Or even worse in January 2023 "a 29-year-old man was attempting to recoup shifted inventory on a pallet inside a trailer. He was trying to move the pallet so that a forklift could move it, but it tipped over and crushed him".

These are only a few of the numerous accidents that occur every day in warehouses all over the world. Forklifts, whose forks are literally weapons, must be utilized whenever goods need to be handled; as a result, the fewer people moving about the warehouse and the less frequently these vehicles are employed, the lower the inherent risk there will be within a warehouse. The risk presented by operators who must scan locations tens of meters high is in addition to the danger caused by using these vehicles and inventorying very high locations. In order to cut expenses and, more importantly, ensure that occurrences like the ones just stated happen less frequently, the trend is to strive to automate operations whenever possible.

In recent years, therefore, new technologies have been considered to carry out this task, in a fully automated way (not involving a human resource) that would be more effective, efficient, and qualitatively superior to carrying out a stock take in a traditional, hand-operated way.

1.1.3 Report on drone usages and business

According to the report [4], start-ups received capital of $7 billion in recent years, of which $17 million was allocated to logistics, $30 million to public administration, and $27.5 million to the healthcare sector. These 3 industries collectively account for 41% of the startups surveyed 1.3. There are also other startups that focus on inspection-related operations, including monitoring of buildings, infrastructure, surveillance, and plants.

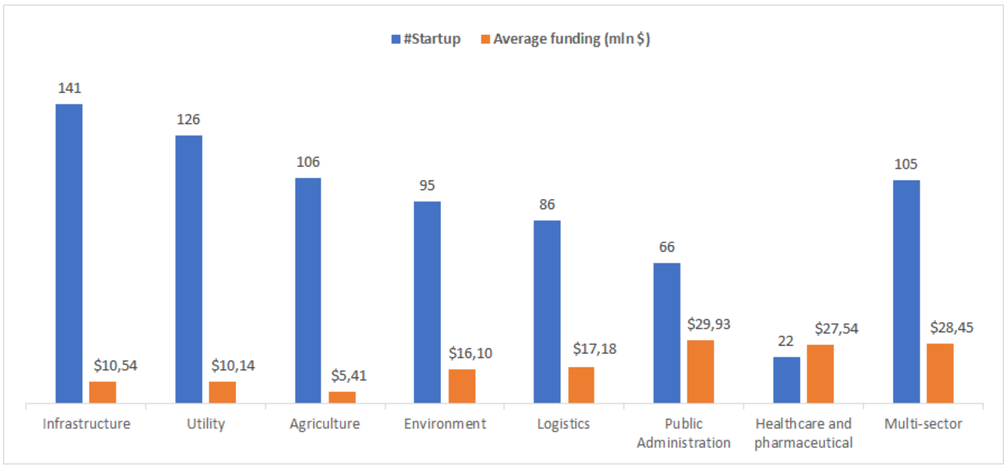


Figure 1.3: Examination sample of 396 startups: the main customer industries of the analyzed startups and their average funding

The year 2021 was a year of recovery for the Italian professional drone market, which reached €94 million with a growth of +29% compared to 2020, in the pre-COVID period investments in this area amounted to €117 million. Indeed, the pandemic situation has highlighted the potential of the technology, which, in the presence of stable regulatory environments, can undoubtedly benefit business and public administration operations. The emergence of numerous start-ups, 317 in 2021 alone, whose value proposition focuses primarily on the development of cutting-edge products, has caused significant ferment on the international front. In terms of applications, 2021 saw a significant increase in projects. In fact, the "Osservatorio.net" [4] examined 228 cases that were implemented in Italy from 2019 to 2021 (30% of the total), 102 of which were implemented in the last year.

The funding made available through the "Horizon 2020" (€80 billion, 2014 - 2020) and "Horizon Europe" (€100 billion, 2021 - 2027) programs, which are among the largest transactional research and innovation programs to fund practical projects in many areas (from healthcare to climate change, etc.), demonstrates Europe's efforts to accelerate the transition to an increasingly digital and sustainable society. A total of €980 million has been allocated to the development or use of drones for

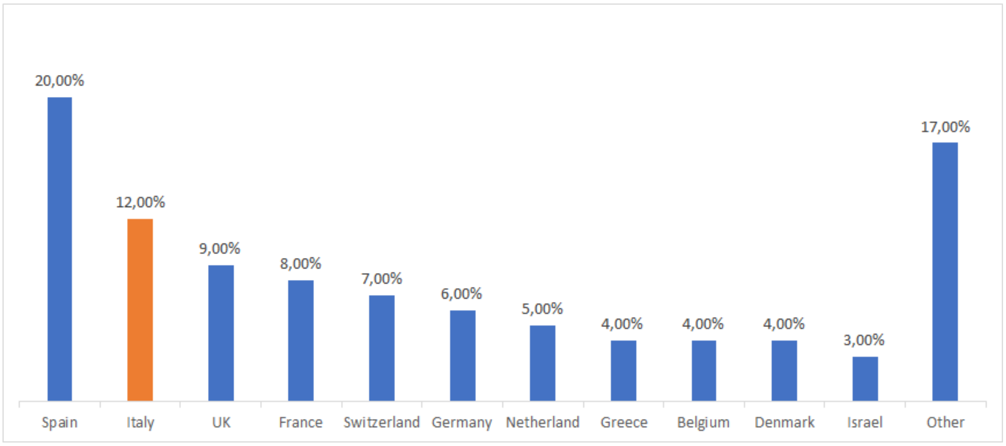


Figure 1.4: examination sample of 320 notices: the category and nation of coordinators of EU drone-related notices

cutting-edge applications across these projects, which include 320 drone-related projects. One of the major participants in "Horizon 2020", Italy, for instance, demonstrates the global interest in drone technology. With 12% of the projects surveyed, it ranks second, as can be observed in the following graph 1.4.

"The drone services market size is expected to grow to $63.6 billion by 2025" [5]

In particular, if we examine the census cases broken down by sector 1.5, we see that there are 97 use cases in logistics, which, when compared to other sectors, emerges as the second largest sector, right after public administration.

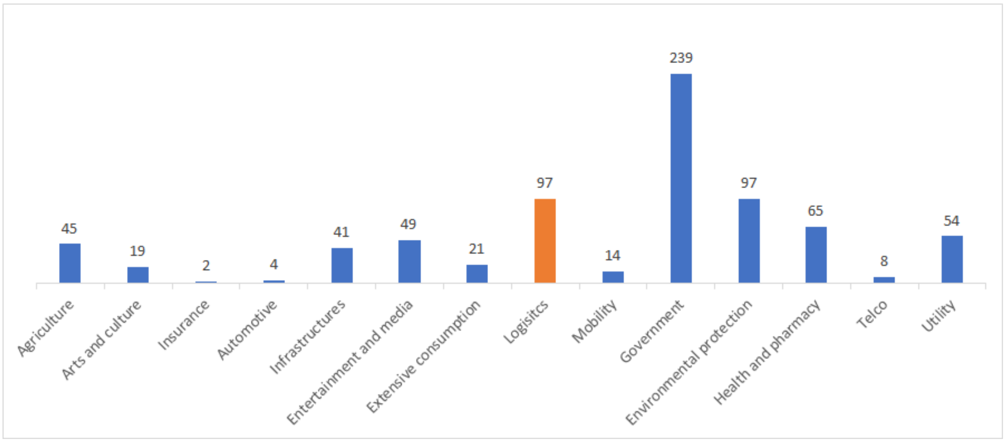


Figure 1.5: Examination sample of 755 cases: distribution of international use cases per sector

Analyzing the logistics industry (see Fig. 1.6), can be observed that the main applications are for the transportation of goods, while there are still few usages for inventory. It is evident that some projects are still in the experimental phase while others are completed and operative.

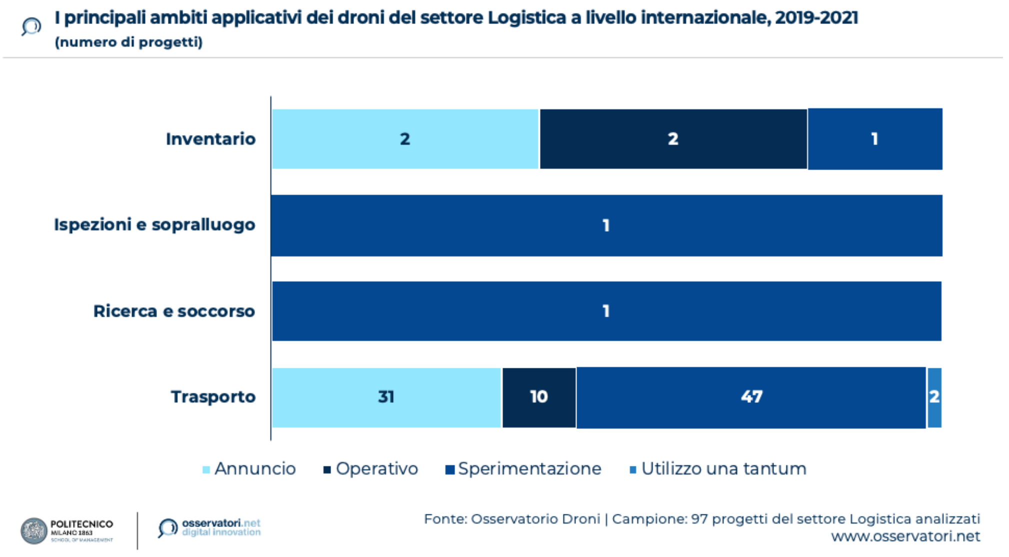


Figure 1.6: Examination sample of 97 use cases: main application domains of drones in the logistics industry worldwide, 2019-2021

1.1.4 Inventory by drone: benefit and challenges

The construction of scalable drone applications is made possible by the onboard processing capability and effective algorithms [6]. Drones due to their ability to fly stably even in narrow environments, their agility, and the capability to hover seem to be very appropriate for this purpose. The main advantage and justification for using drones for warehouse inventory is that they can fly indoors, maneuver deftly through warehouse aisles, and reach great heights with no issues. Drones can be moved considerably more quickly and effectively than operators can, especially when moving them at heights. In this way, it is possible to navigate and "visit" all the locations within a warehouse very effectively and quickly. Once the drone is in front of a specific location (we will see later the different techniques to retrieve this data) the drone takes a photo and analyzing it through AI model and image processing algorithm it is possible to understand what products are there, their status, scan the present barcodes, etc. By matching the information extracted from the photo with the location data where the photo was taken, the actual inventory is taken. At this point the location is considered "closed," and the results of this scan process can be sent to the WMS. This process replaces an operator scanning and visually analyzing the status of the goods and then manually entering the data on the WMS, so as you can imagine the advantage is tangible. Of course, the benefit of using such a method is enormous for very big, vertically constructed warehouses.

There are limitations imposed by the implementation of a drone program due to the variety and complexity of warehouse structures. Physical location, the typology of items stored, the layout (such as shelves, pallets, and boxes), size, and technology are all different. Furthermore diversified is the purpose of warehouses. For instance, a distribution warehouse functions differently from a cross-docking warehouse and a production warehouse for raw materials and finished goods. [6]

Only a few warehouse scenarios make sense and justify using drone-based inventory since it may not be realistic to consider this type of solution in every warehouse, for instance, the warehouse must have the following features:

- High racks ($>$ 5 meters) - Dangerous tasks for operators to be exposed to high altitude. For shelves lower than 5 m thus solution doesn't make sense
- Relatively large size $>$ 10000 square meters
- Single deep pallet rack - Barcode scanning not possible for double-deep storage principle
- Long corridors ($>$ 50 meters) - Long walking distances increase the time needed to accomplish tasks

The most challenging aspect of using these objects inside a warehouse, however, is the difficulty to localize and to position (the drone pr system) within an indoor environment; as it is not possible to use GPS, which is today the primary method of localization.

One of the crucial elements of this kind of application is localization, which needs to be extremely precise, on the order of cm. While mistakes can actually be tolerated in outdoor applications up to a few meters, this is not true for inside applications, where errors of even a few centimeters could result in collisions with obstacles or system failure, such as the system getting stuck and unable to move. This feature is therefore turned worse in the industrial scenario: suffice it to say that a drone error of even 10 cm could result in a mistake in the location-reading code association. Because the drone believes it is facing a specific location while it is actually above or below it, it would result in an entirely inaccurate inventory. Luckily to date, one of the most advanced visual SLAM algorithms achieves an accuracy of 5cm [6].

As we saw in the previous example, when the inventory findings are submitted to the WMS, they will be completely wrong and result in errors that, when they occur on a large scale, will be catastrophic and cause millions of dollars in economic damage.

Therefore, it is necessary to use other indoor localization and tracking systems that use different technologies to be able to orient, localize, and then navigate properly. We will delve deeper into these aspects throughout this thesis.

Drones can be used for a variety of inventory management functions, including auditing and managing inventories, cycle counting, finding specific items, maintaining buffer stocks, and taking stock. "Inventory management applications appear to have the highest potential for use in warehousing operations: 7 of the 12 use cases fall into this category"[6]

1.1.5 IKEA use case

Today there are already companies, including very important and famous ones, that have adopted cutting-edge technologies to improve, automate and optimize operations within the warehouse. A good example is IKEA, which "is investing in technology across the board so that our stores can better support customer fulfillment and become true centers for omnichannel retailing. Introducing drones and other advanced tools — such as, for example, robots for picking up goods — is a genuine win-win for everybody. It improves our co-workers' well-being, lowers operational costs and allows us to become more affordable and convenient for our customers"[7].

Like any other warehouse, they have problems with discrepancies between the digital data and the actual inventory.

A typical IKEA shop has about 7000 shelf placements, whereas distribution hubs have ten times more. The accuracy was not adequate despite efforts to improve data quality. Ten full-time employees of a distribution center spent their entire workday looking for misplaced pallets.

IKEA has historically taken manual running inventories, primarily before and after opening and closing. Employees must lift a pallet down, count its contents, and then lift it back up again. This is a very monotonous and tiresome task. Imagine repeating this 7000 times for a store, and 70000 times for a warehouse.

Eight drones are required to scan every site in a store with 7000 pallets in a single day (Sunday, when the store is closed). On other days, the drones only fly at night and after hours, and they only inspect places where the data indicated a material movement. Any data mismatch is fixed either manually or digitally after the flights. It seems that algorithms can be very helpful in finding deltas. As a result, IKEA sites using Verity [8] drones have daily data accuracy that is close to 100%. [9]

Since 2021, IKEA's Swiss division has also started adopting drone-inventory technology. According to the furniture company, 100 Verity drones are currently in use across 16 sites in Belgium, Croatia, Slovakia, Germany, Italy, the Netherlands, and Switzerland performing more than 300000 inventory checks per month.

1.1.6 Other drone application in logistics

As it is reported in [6] "Drones have shown high potential in the logistics industry. It has been speculated that this market will grow by $29 billion by 2027 with an annual growth rate of almost 20%"

Drones can also be used in intralogistics. They can, for instance, transport components from locations of storage to manufacturing operations. There is a lot of promise for indoor use for drones that can carry payloads and follow specified flight paths, such as for on-site quick delivery of tools and replacement parts. For tasks requiring monitoring and inspection at great heights or in hazardous areas, indoor drone use is suitable. Drones can be used for regular surveillance tasks to stop stealing and other illicit conduct.

1.2 Thesis organization

1. **Insight into the use case of autonomous inventory and logistics applications**

   In-depth exploration of stocktaking and other operations that can be carried out autonomously in a warehouse based on drones or robots. Specifically for warehouse inventory, there are manual, semi-autonomous, and completely autonomous solutions, and in this case, we will focus on different solutions with different approaches.

   - *Manual or semi-automatic drones:* These are usually consumer drones (with limited sensor and computational capabilities), such as DJI drones, for which ad hoc software is developed. In the manual version, the drone is piloted by an operator who moves it around the warehouse to take pictures, etc. Then in the semi-automatic solutions, the drone can move semi-autonomously within a warehouse, but only after it has been manually positioned at a specific location in the aisle. At this point it is able to perform some operations "autonomously"; however, in these cases, the operator must always be ready to take control in case of problems as the drone is not able to make decisions autonomously
   - *Standalone drone and swarm of drones:* They are customized drones with specific and dedicated hardware (cameras and sensors) on board that enable them to navigate autonomously in indoor environments. They can work stand-alone or be organized into fleets to parallelize operations
   - *Drone supported by AMR:* The system consists of a drone that is supported by an AMR to localize itself: the AMR on the ground locates and navigates itself, while the drone tracks and follows it. In some cases, the AMR

can also provide power to the drone, so the overall battery life is greatly increased.

- *AMR with multiple cameras mounted on a pole:* Other systems, however, involve the use of an AMR with a pole (fixed or telescopic) installed. Cameras are mounted on this pole, which similarly to what the drone does, take pictures of the highest locations

2. **State of the art for indoor navigation**

   A brief overview of indoor navigation techniques such as SLAM (Simultaneous Localization and Mapping) and indoor localization technologies such as UWB (Ultra-Wideband), wheel odometry, VIO (Visual-Inertial Odometry), fiducial marker relocation, and the significance of obstacle identification and avoidance. Furthermore, sensor fusion and the use of the Extended Kalman Filter are explored as essential components for achieving accurate and robust navigation in indoor environments.

3. **Empirical studies and measurements for sensor fusion optimization**

   This chapter focuses on a variety of project-related topics, beginning with the hardware and software setup. Different ROS (Robot Operating System) nodes are implemented to facilitate communication between data from VIO, camera streaming images, and fiducial marker data: all together to compute the sensor fusion. Briefly, some "publisher" nodes produce data (from cameras or sensors) while other nodes "subscribe" to the publisher to get and process these data in order to apply some logic and make some decisions. Although sensor fusion techniques(multiple nodes take data from different producers and work together to produce a unique final result) are being investigated to increase localization accuracy, preliminary measurements show that there is a problem with fiducial marker relocation. In order to undertake a full investigation of the fiducial marker relocation, four different cameras were employed while AruCo Marker and AprilTag Marker were used in the comparison of fiducial marker families. To achieve precise measurements, camera calibration techniques are used; this is a highly crucial and significant phase. Finally, the results obtained from the experiments are presented, highlighting some differences between cameras and fiducial marker families.

4. **Conclusion** In this last chapter, all the obtained results are presented, starting with the preliminary outcomes of phase 1 and then delving into the comparative analysis of camera and marker performance in phase 2. It becomes evident that, overall, AprilTags have superior performance compared to ArUco markers. For our specific use case, the D435i camera appears to be the most suitable for fiducial marker relocalization. It boasts good resolution, maintains a high

frequency in the data output stream, and is stable in marker identification, making it a well-balanced choice. Additionally, its lightweight and low power consumption make it suitable for installation on a drone in this kind of application. However, for effective relocalization, it is crucial to design marker dimensions having in mind the effective work range of distances for marker reading. In any case and precision is maintained when the distance between the camera and the marker is not excessive. In any case, regardless of the marker's size, the accuracy and stability of the estimated pose are strongly dependent on the distance between the camera and the marker. While a larger marker can partially compensate for a greater reading distance, relocalization for precise results still needs to be performed within a certain threshold distance. Finally, a concluding section provides insights and ideas for possible future work.

Chapter 2

Insight into the use case of autonomous inventory and logistics applications

2.1 Operation inside a warehouse: companies wants automation!

"100,000 Automated Guided Vehicles (AGV) and Autonomous Mobile Robots (AMR) were shipped globally in 2021 alone. In terms of revenue, AMRs generated $1.6 billion and AGVs grew to $1.3 billion" [10]. And these markets are rapidly growing, with for AGVs and AMRs projected to grow to $20 billion by 2028 [11].

Daily operations within a warehouse ensure its proper management and operation: the more optimized a warehouse is, the greater its performance and effectiveness, and the better service customers/users will experience. The number of orders that the entire system can fill in a given amount of time is used to measure a warehouse's performance; the higher this number, the more effective the warehouse will be. Human resource effectiveness, or how many operators are required to complete a given task or set of tasks, is another factor to take into account. The more effective and optimized their work is, the less stuff I will need to complete the same amount of work, as you can imagine. Extremizing this idea to the utmost, big companies today try to optimize, make it efficient, and automate as many warehouse processes as they can. In fact, by automating some processes, it is possible to significantly improve performance and lower costs.

Therefore, the current trend is to try to automate and have robots perform all those labor-intensive, alienating, time-consuming, repetitive, dangerous, and easily error-prone operations where a human being can add zero value. Inventory management, massive picking operations, inbound, outbound, put-away operations, etc. are a few examples. In many of these scenarios, highly automated systems support or replace the human operator, optimizing these operations and allowing him to focus on other tasks where his added value is maximized.

For massive picking and put-away, there are AMR robots with a payload mounted that basically consists of n totes: this robot is able to autonomously navigate within a warehouse and pick up the plastic and carton parcels and take them to the packing area to then be packed and shipped.

It is also able to perform the opposite operation i.e. once it receives all the goods, the robot is able to sort them and place them in proper storage on the shelves. The telescopic model of this system is also able to reach considerable heights of about 12 m.

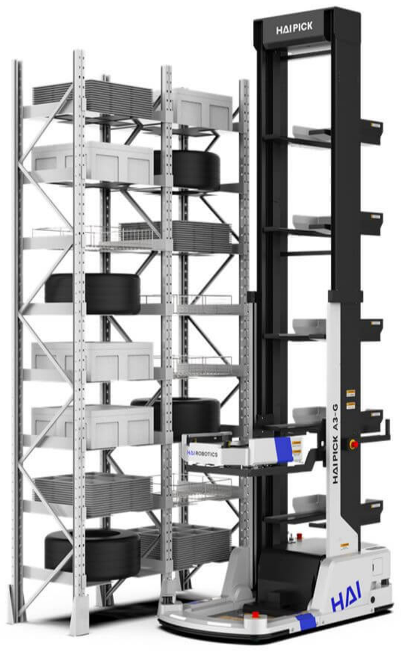


Figure 2.1: Example of autonomous picking and putaway robot

An indirect consequence of adopting these systems is that in addition to improving the performance and effectiveness of processes, they are also able to optimize space, increasing density and thus making the best use of space and footprint in the warehouse, further reducing costs. Since there are no more constraints on the "ergonomics" of the spaces, to allow humans and forklifts to move within the aisles of the warehouse, it is possible to reduce the width of the aisles to the minimum necessary to allow the robots to operate, thus making the best use of the space available.

Taking the concept of automation to the extreme, we advance from partially automating specific operations and processes within a warehouse to completely automating the entire warehouse. Fabric [12] and Autostore [13] are just two examples of these kinds of solutions. Autostore is an automated storage and retrieval system (ASRS), it harnesses the power of warehouse robots for 24/7 order fulfillment within a cubic layout so dense it can actually quadruple storage capacity and unlock the true potential of storage floor space. It can increase the storage capacity 4 times, 99.7% Uptime 24/7 access to inventory (given for free, transparently way as the system is totally autonomous).

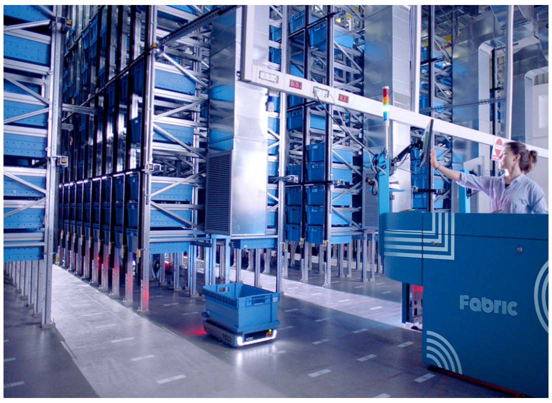


Figure 2.2: Fabric: retail fulfillment automation company

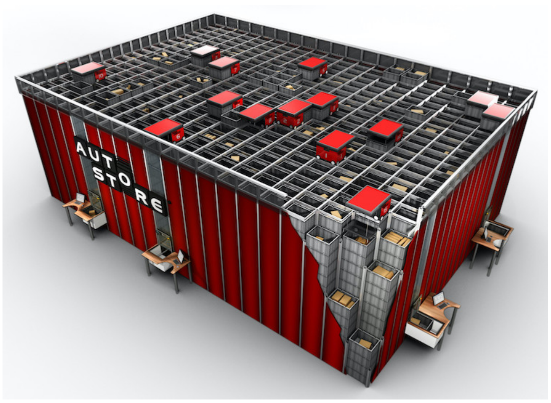


Figure 2.3: Autostore: an automated storage and retrieval system

2.2 What role does inventory play in a company's success?

Warehouse and supply chain efficiency are directly related to inventory management. It entails managing the movement and storage of goods within a warehouse, making sure that stock is available when needed, reducing costs, and maximizing overall productivity.

Inventory management is a critical component of warehouse operations, offering numerous benefits to businesses. First off, it gives warehouses the ability to quickly satisfy client needs, ensuring timely order fulfillment and enhancing customer satisfaction.

By minimizing understock[1], stockouts[2], overstock[3] and deadstock[4], using specific strategies based on consuming first seasonal and expiring products, businesses can

[1]Underswtock: to stock with less than the usual or desirable number or quantity

[2]Stockout: also know out-of-stock (OOS) is an event that causes inventory to be exhausted. There are no goods of a particular kind available for sale.

[3]Overstock: also called "surplus stock," happens when stores purchase more products than they sell, this leads to having too much stock in a warehouse that has not sold which increases storage costs and reduces working capital.

[4]DeadStock: an amount of a product that a company has bought or made but that is no longer sellable and will likely never sell in the future, oftentimes because it's expired, obsolete, low quality, or out of season. Storing dead stock costs money and so reduces the amount of profit a company can make. Dead stock only refers to inventory that has never been sold, which excludes returns.

lower costs and maximize their capital efficiency. Businesses can strike a balance between maintaining adequate stock levels and preventing needless holding costs by implementing inventory control strategies, optimizing replenishment processes using precise consumption forecasts, and choosing which product to promote or discount.

Inventory management's effect on supply chain effectiveness is another essential factor. An efficient warehouse inventory management system enables more seamless coordination of the activities of production, distribution, and procurement. Businesses can reorganize their processes, reduce disruptions, and enhance the performance of their supply chains by having accurate and timely visibility into inventory levels. In some circumstances, it is also possible to take advantage of the inventory phase in order to assess the quality of the stocked goods.

2.2.1 Different typology of inventory counting

It takes a lot of time and manpower to conduct a thorough inventory of every item in a business, including the sales floor and backroom.

This is the obvious option if the warehouse is closed once a week. Larger businesses, however, can frequently run seven days a week. Retailers are then faced with another challenging decision: stop the warehouse or carry out the stocktake over the course of the night? Obviously, this compromises operations, and performances that led to upsetting consumers. While having to complete the inventory count overnight can cause a schedule crunch. When employing a third party, an overnight inventory count is more practical (and popular), but it is not always possible if you are employing internal workers.

The truth is that manual inventory counts produce mediocre outcomes. There will still be errors and discrepancies even with supporting technologies like barcode scanners because there is a crucial margin for human error. There are several causes of inconsistencies, including damage, theft, incorrect amount entry, incorrect item or location scanning, shifting products between shelves without updating the system, etc. Doing stock counts once or twice a year is no longer sufficient, which is a significant issue. You might have 90% accuracy once every six months, but what about the rest of the time? The average accuracy of store inventories is between 60 and 80 percent.

Cycle count Cycle counts are an alternative to annual stocktakes. When a store conducts a cycle count, it counts only a portion of its inventory. It is possible to carry out a cycle count many times a quarter rather than an annual stocktake once or twice a year. In essence, you are segmenting the annual count into a number of mini-stocktakes based on product categories, product groups, or warehouse/store zones.

This has many advantages over the annual stocktake, including fewer disruptions to the store, less labor-intensive work, and lower costs (because retail staff may complete it instead of employing third-party manpower). Additionally, it may reduce the noticeable variance that occurs when there is a longer interval between takes. Instead of concentrating all of your efforts on that one weekend, you may spread out the labor across the entire year. Planning cycle counts is simpler, and your ERP (Enterprise Resource Planning) can even generate counting orders for you based on your rules and stock changes.

Planning is essential in this situation because there is always a chance of missing items when constructing the counting orders when not counting everything. Another drawback is that counting becomes ingrained in your employees' daily routines, making it simple for them to become sidetracked (for example, when a customer approaches them) and either forget to count anything or count it twice. As cycle counting could be happening on a daily basis, it may force you to do it during open hours, but in this scenario, there is always the possibility that items included in the counting orders could be sold or shipped before being actually counted.

ABC Analysis Cycle Counting ABC analysis cycle counting relies on a class system. In order to differentiate between Class A, Class B, and Class C items, the inventory manager or inventory control system performs a statistical analysis based on a number of different criteria. The A items are counted more frequently than the B items which are counted more frequently than the C items. Depending on what's crucial to the warehouse, several factors are employed to categorize goods. Generally speaking, there are three different types of ABC analysis cycle counting:

- Pareto Principle-Based ABC Analysis: The Pareto Principle approach, also known as ABC cycle counting, makes the assumption that 20% of a warehouse's inventory corresponds to 80% of its sales. These are the "A" items (the "B" and "C" products account for 30% and 15% of the inventory, respectively, and so on). Your most valuable assets or fastest-moving SKUs might be "A" goods. The counted as A, B, or C items can be distinguished by inventory control software. Consider counting your "A" things more frequently while counting your "B" and "C" goods less frequently.

- Usage-Based ABC Analysis: Items are more likely to get lost the more inventory is moved around. As a result, several warehouses prefer to employ a usage-based method for their ABC analysis. They rate their inventory according to how frequently it moves, rather than keeping expensive things in Class A exclusively. Therefore, even if a product isn't very expensive, it will be counted more often if customers buy it regularly.

- Hybrid ABC Analysis: This strategy combines automated statistical analysis and experiential modifications, the best of both worlds. The hybrid method uses the Pareto Principle as a starting point to identify the supply chain components that are producing the most value. This division of the available goods is not fixed. Instead, based on their observations, the supply chain management team can change the classifications.

Control Group Typically, the procedure focuses on a small subset of items that are counted repeatedly in a brief period of time, revealing any faults in the counting approach (which can subsequently be rectified).

Random Sample Just as it sounds, a certain number of things are randomly chosen to be counted. Daily counts allow you to quickly account for a significant portion of the warehouse's inventory.

Process Control Cycle Counting With this approach, counters can select which portions to count. Counters undertake a count if there are any concerns after reviewing the existing inventory record for any discrepancies. Process control counting is "controversial in theory, [but] effective in practice," according to academic research on the subject. This is due to the fact that, at first look, employees can simply choose the easiest inventory to count and skip difficult inventory by relying on the existing record, if they wish. In practice, a review of the method found that while statistically the process control method is biased, in practice it's biased towards inventory items with the highest chance of inventory inaccuracy, and these are the sections a warehouse wants to focus its attention on anyway.

Ad-hoc/Opportunity-based Cycle Counting Ad-hoc counting is frequently started by the user and is not planned, making it useful in unusual circumstances. Let's say someone counted a zone, and a few days later the system malfunctioned because of someone or some process. An opportunity-based cycle count may be performed in the following situations: when an item is reordered; when an item is stowed; and when an item's balance falls below a predefined threshold. Instead of waiting for the next cycle count, you can just construct a new, empty order and begin adding counted items to it. Of course, your system must be able to compare the quantity that was tallied to the data stored in the software. Ad-hoc counting is sometimes referred to as spot or blind counting because it typically takes place in unscheduled, tiny store/warehouse zones. You have more flexibility with ad-hoc counting to handle emergencies and get around the problems caused by bad cycle count planning. Ad-hoc counting analysis and results may enable you to precisely modify your cycle count plan. The difficulty in this situation is the same as the benefit, which is the human element. Even when they recognize a need for it, if your staff are not cautious enough, they might not initiate ad-hoc counting.

Tag counting In preparation for tag counting, the store/warehouse employees should place a physical tag on each item. The operator must complete the required slots on the tag with the item ID, counted quantity, and other pertinent information during counting. Some tags have two sides, allowing a second worker to check the data and, if necessary, fill in the adjustment on the other side. These tags are gathered and added to the system as journals after the counting procedure is complete.

The primary distinction between tag counting and all other counting methods is that tag counting does not involve a direct comparison to system data. As opposed to just generating a counting order and comparing it to the tag counting list, it is more like creating a rough draft list of your goods and quantities, polishing it (e.g., updating with sales that have occurred during the counting), and then comparing it to the final list. Tag counting is a great option for bulk warehouse zones or retailers, which frequently lack spaces suited for a true warehouse. When there are numerous objects and no places, keeping track of the counting procedure is challenging. In this case, tags provide visual information about what is counted and what is not, resolving the problem.

Utilizing scanners while counting is an even more effective option. The item can then just have the physical tag attached so that it can be recalled after it has been counted after the tag ID, item, and quantity have been scanned. There will be no need to manually enter the journals; they will be created right away.

RFID Using product tags that emit a small radio frequency, shops may quickly count their inventory by exploiting a technology called radiofrequency identification (RFID). A single employee with an RFID reader can count thousands of products, enabling stores that have adopted RFID to conduct weekly or even daily stocktakes in a few minutes. As a result, RFID stores typically have 99 percent accuracy in their inventory and no longer require annual stocktakes.

RFID installation in businesses is a large endeavor that costs money upfront. The expense of doing annual third-party stocktakes, on the other hand, could be utilized to pay for the RFID project since it offers a better return on investment than traditional physical inventories.

Conclusion Finally, warehouse inventory management is essential for satisfying customer needs, minimizing expenses, and improving supply chain efficiency. By maintaining ideal inventory levels, businesses can guarantee timely order fulfillment, save costs associated with overstock, and streamline their entire operations.

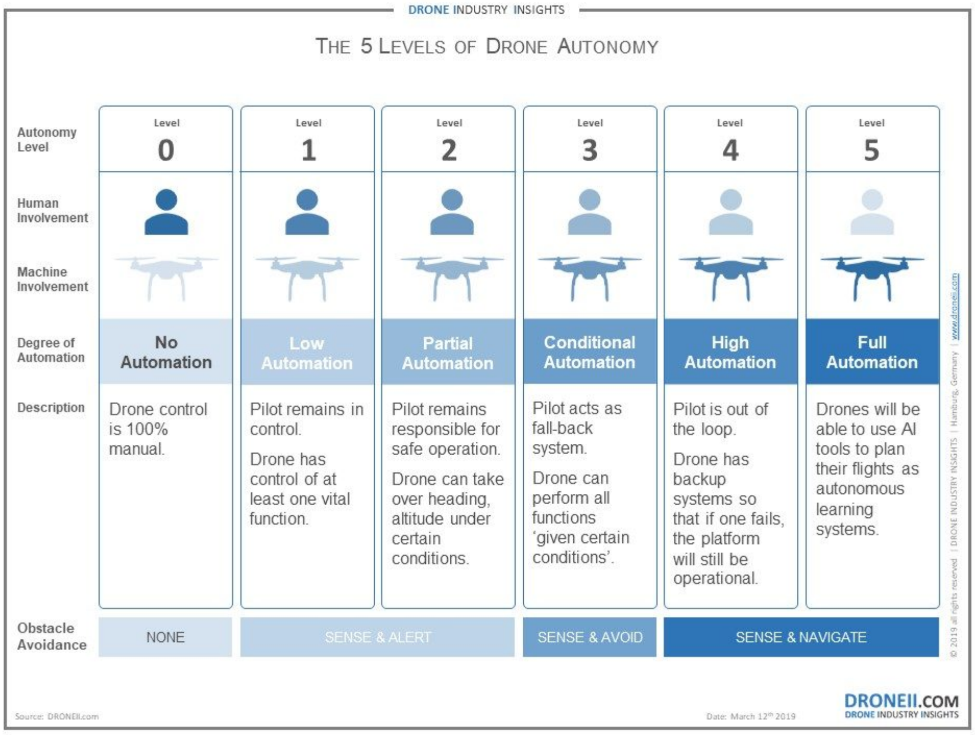
Implementing efficient inventory management procedures increases profitability, boosts customer satisfaction, and gives businesses an advantage in the marketplace.

2.3 Automation levels

First, let's clarify the difference between automatic flight and autonomous flight.

Automatic flying: operations in which the remote pilot must be present and ready to intervene at any time if needed. This is allowed only for the C4 class (under 25 Kg, subcategory A3 - fly far from people), while it is not explicitly prohibited for all other classes in the Open category (C0 to C3, the 'open' category section is the main reference for the majority of leisure drone activities and low-risk commercial activities).

Autonomous flight: operations in which an unmanned aircraft operates without the remote pilot being able to intervene. This is not permitted in the Open category, while it is permitted in the Specific and Certified categories subject to authorization by the appropriate national authority.


DRONE INDUSTRY INSIGHTS

THE 5 LEVELS OF DRONE AUTONOMY

Autonomy Level	Level 0	Level 1	Level 2	Level 3	Level 4	Level 5
Degree of Automation	No Automation	Low Automation	Partial Automation	Conditional Automation	High Automation	Full Automation
Description	Drone control is 100% manual.	Pilot remains in control. Drone has control of at least one vital function.	Pilot remains responsible for safe operation. Drone can take over heading, altitude under certain conditions.	Pilot acts as fall-back system. Drone can perform all functions 'given certain conditions'.	Pilot is out of the loop. Drone has backup systems so that if one fails, the platform will still be operational.	Drones will be able to use AI tools to plan their flights as autonomous learning systems.
Obstacle Avoidance	NONE	SENSE & ALERT		SENSE & AVOID	SENSE & NAVIGATE	

Figure 2.4: The 5 levels of drone autonomy

Autonomous flight is necessary when there are communication difficulties, when decisions are time-critical, or when they can be made in a better way using data available on board the drone. It is also attractive for reducing remote pilot workload and system-level complexity, with the aim of improving the robustness and performance of the system itself. Building on this, JARUS (Joint Authorities for Rulemaking on Unmanned Systems) together with ENAC (Ente Nazionale per l'Aviazione Civile) is developing performance-based requirements for the certification of UAS that include functions implemented through highly complex systems equipped with Artificial Intelligence, Machine Learning, Deep Learning and neural networks. These requirements will make it possible to certify certain autonomous functions of drones.

A document from JARUS has been released to provide a common framework for tackling the implementation and effects of progressive automation of functions. The benefits of automation have long been recognized by the aviation industry, which has long used it. In addition to optimizing ground operations to cut costs and increase efficiency, automating some tasks frees up pilots to concentrate on more crucial aspects of the flight. The term "autonomy," however, has become more frequently used in relation to UAS, which led to misunderstandings.

Before analyzing each automation's level, it is necessary to understand the following concepts as defined by JARUS [14]:

The Human-in-the-Loop (HITL): Is a method for managing system parameters that directly involves humans in providing inputs and assessing outputs. With this method, people can actively participate in system operation and be a crucial part of the feedback loop.

Human-on-the-Loop (HOTL): is a method for system control where a human supervises a device that generates inputs and assesses outputs to control system parameters. Contrast this with the Human-in-the-Loop (HITL) approach, in which a human directly provides inputs and assesses outputs to control system parameters. In the HOTL method, the human monitors the system's operation rather than directly controlling it. The human is in charge of ensuring that the system operates properly and safely, while the machine is in charge of making decisions and performing tasks. As the human can intervene and assume control if necessary, the HOTL method adds an extra layer of safety and reliability to the system. This makes it possible to create systems that are more complex and autonomous while still guaranteeing that a human is involved in the decision-making process.

Human-off-the-Loop: Refers to a method for system control where system parameters are not monitored or managed by humans. Instead, a machine is in charge of supplying inputs and assessing outputs to ensure sure the system operates properly. In highly automated systems, where the machine is capable of operating independently without requiring human intervention, this type of control is frequently used.

Operational Design Domain (ODD): is a mechanism that helps define the operational boundary within which a particular system or function has been designed to operate.

Knowing this vocabulary it is possible to understand the automation levels defined by Jarus [14]:

- **Level 0 – Manual Operation:** The human manually executes the function, receiving no support from the machine.
- **Level 1 – Assisted Operation:** Functions at this level of automation are designed to assist the human in performing tasks. The machine operates in a supporting role outside the loop of human actions. Although the human is still in control of executing the function, the machine can provide limited assistance within the designated ODD, such as providing relevant information.
- **Level 2 – Task Reduction:** This level of automation involves shared control and monitoring between the human and machine, where the machine takes on an in-the-loop management role to help reduce the human workload and/or skill level required to complete the task. Although the human still leads the function's execution, the machine now provides a more substantial level of support within a clearly defined ODD.
- **Level 3 – Supervised Automation:** At this level of automation, the machine performs the function while the human supervises and can intervene if necessary. The human is not aware of the machine's internal states but supervises the outcomes for safety. The machine leads the execution within a defined ODD, but the human continuously monitors and must have the necessary information to intervene if needed. Careful human factors system design is required to ensure that the human has all the required information to transition from "on-the-loop" to "in-the-loop" when necessary.
- **Level 4 – Manage by Exception:** At this level of automation, the machine performs the function independently and alerts the human only when an issue arises. Unlike lower levels, the human is not required to monitor the function in real-time, but must be available and able to intervene if needed. Once the machine has proven its ability to perform the entire function effectively and respond to the environment, the crew may trust it to operate without human supervision within a specified ODD. Building trust requires ensuring the system's trustworthiness, including meeting safety expectations for reliability, integrity, and assurance.

- **Level 5 – Full Automation:** In a fully automated function, the machine assumes full responsibility for executing the task, while the human's understanding of operational parameters is minimal or non-existent. The human's interaction with the machine is usually limited to providing strategic directives, such as pre-flight planning, and observing the outcomes. Additionally, without special authorization, humans cannot intervene in real time due to practical limitations or deliberate exclusion within the ODD. Such operations are expected to require advanced technologies, such as Artificial Intelligence, or strict limitations on the ODD to restrict the autonomous function's operation.

In figure 2.4, you can find a scheme of the presented levels.

2.4 Manual-traditional inventory

Traditional stocktaking methods, which rely on manual or paper-based processes for tracking and managing inventory, have both advantages and disadvantages that warrant consideration.

On the positive side, these methods benefit from familiarity. Having been used for many years, warehouse personnel are often well-acquainted with these processes. This familiarity reduces the learning curve and minimizes the need for extensive training, allowing for easier adoption and implementation.

However, traditional stocktaking methods also have their drawbacks. One notable disadvantage is the time-consuming and time-intensive nature of manual processes, particularly in larger warehouses. Physical counting of inventory items and repetitive manual data recording can be labor-intensive and prone to errors. These errors can result in delays and necessitate additional efforts to reconcile discrepancies, consuming valuable time and resources. In addition, expensive equipment, like scissor elevators, is needed for vertically developed warehouses. Personnel must also be trained and specialized to operate this parcel at such high altitudes (up to 20 m). In addition to the expense of the equipment, this method of inventory at heights is also very risky because, despite the operators' training and specialization, they still run the inherent risk of falling because of their height.

If we consider that during a traditional inventory we frequently do not have real-time visibility into what is happening and inventory levels, if not only after inventory is completed, perhaps hours or days later, it is also necessary in some circumstances to stop other operations in the warehouse areas where the stock take is being performed. This lack of real-time information poses challenges when it comes to responding promptly to changes in demand or unexpected events. The absence of immediate visibility hampers decision-making and reduces the agility of supply chain operations, potentially resulting in missed opportunities or suboptimal outcomes.

Considering these pros and cons, businesses need to carefully assess their specific needs, resources, and goals when deciding which stocktaking methods to employ. While traditional methods offer familiarity and affordability, they also present challenges related to time consumption, accuracy, and real-time visibility. Exploring alternative inventory management solutions, such as technology-driven systems or automation, may offer potential improvements in efficiency, accuracy, and decision-making capabilities.

2.5 Semi-autonomous inventory

Between the traditional manual and fully autonomous solutions, there are intermediate solutions: the semi-autonomous ones. These solutions, as can be deduced from the name are characterized by having a partial level of autonomy; the contribution or supervision of an operator is also required. For example, some solutions consist of a drone manually piloted by a trained operator or a drone capable of autonomously conducting an inventory of only one shelf at a time.

To conduct a fully autonomous inventory, the warehouse must meet several requirements, often very restrictive, and therefore a fully automated solution cannot always be deployed either because of technical infeasibility or due to costs that are too excessive. This scenario provides a suitable balance because in certain situations the financial commitment could be lower than in a completely automated system, where the initial expenditure is sometimes very high. However, some semi-autonomous systems require the warehouse to be particularly prepared, set up, and compliant with a number of requirements (less stringent than a completely autonomous warehouse) in order for them to work.

Finally, it should be noted that due to certain warehouse characteristics, it is not always possible to have fully autonomous solutions. For example, in cases where the layout of the racks is dynamic and can change on a weekly basis, it is impossible to map the work area and enable autonomous navigation. In these situations, a semi-autonomous solution that includes manual guidance may be the best option.

2.5.1 Possible solutions to semi-autonomous inventory

Let's see now in detail how the solutions mentioned in the previous chapter work. In both solutions, the association between location and pallet is made when the QR code of the location and the label of the pallet in that location are displayed in the same frame. The biggest drawback of this approach is that it requires advance preparation of the entire warehouse because each site needs to have a unique QR attached to it that univocally identifies it. The WMS then receives the photographs the drone took, allowing for the viewing of the product's quality condition as well as a history of the locations.

The operator of the drone (a commercial DJI) in the first solution is able to scan and inventory even the highest floors without putting himself at risk and, more importantly, doing it fast and efficiently without using any additional equipment.

The second approach uses a drone that can fly by itself, but only in a single aisle; in this case, operator supervision is still necessary to ensure the drone is flying correctly: the operator positions the drone at the takeoff point before inventorying a single shelf. As a commercial drone (DJI) without specific sensors for autonomous navigation is being employed, computer vision is the main technique of navigation. The proximity sensors that it is equipped with are disabled and unable to be used due to the narrow aisles.

Once the drone is positioned at the take-off point, the operator initiates the inventory mission: the drone takes off and searches for the QR code reference of the first known location in its map (grid).

The drone uses computer vision to track the QR code of the location as a reference and centers itself before taking the first picture. The drone then receives movement commands on the horizontal and vertical axes (pulses of acceleration of a given time interval), allowing it to move horizontally and vertically from one location to another. The only way the drone can orient itself and correct its position after moving is to trace the QR of the location it should be in front of after moving, then center itself again if the inaccuracy is larger than a specified tolerance threshold. The drone's movement has no feedback; in fact, it moves "blindly".

Similar to this, if the QR's dimensions reveal that it is too close or far from the shelf, it will automatically adjust its position by moving closer or farther. The drone returns to the takeoff point once all the desired locations have been scanned. The operator then picks up the drone and places it on the new takeoff point of the new shelf to be scanned, or changes the battery if it is running low. Since the drone's proximity sensors are disabled, it won't be able to detect any obstacle in its trajectory. As previously noted, proximity sensors are deactivated, so any objects in the drone's trajectory won't be detected. As a result, operator supervision is still necessary to examine the area, remove any obstacles from the drone's path, and ensure a safe landing.

2.6 Autonomous inventory

Fully autonomous inventories provide the highest level of inventory optimization and effectiveness. In this scenario, the operator only needs to schedule and plan an inventory, start it, periodically check its progress, and read the final results after the inventory is finished. There are requirements that must be satisfied for automation to function correctly, including the warehouse's shelves, aisles, floor, lighting, etc. In the next section, we are going to investigate a few examples of systems that conduct inventories independently as well as the technology employed in these systems.

2.6.1 Possible solutions to autonomous inventory

In this section, we are going to discover different types of approaches and solutions to perform inventory operations within a warehouse autonomously. We will see the main features advantages and disadvantages of each solution. The 3 approaches we will focus on are standalone drone, AMR-supported drone, and AMR with a telescopic pole with cameras mounted on it.

Let's analyze different solutions involving the use of a standalone drone: as can be easily imagined, the main limitation of this solution compared to a solution supported by an AMR is the flight autonomy, which hardly goes beyond 20 minutes, compared to the other solution that can have autonomy of even several hours. Without the ability to use GPS to locate and orient itself, a drone attempting to navigate autonomously within an indoor environment (such as a warehouse) must rely on other techniques and technologies. These methods and technologies are frequently combined for the sake of robustness and redundancy and are then fused together by a Kalman filter that assigns the appropriate weight to each input data sensor, based on situation and circumstance.

2.6.2 The used technologies

The technologies and techniques that can be used to enable a drone to autonomously navigate an indoor environment are: Lidar, UWB (Ultra Wide Band) antennas, depth and sensor chambers such as to enable Visual Odometry, and finally, the use of tags and markers (e.g. Fiducial marker) that ensure a reliable relative reference in space.

Lidar - light detection and ranging:

It is a laser that rotates at a very high frequency and allows scanning and mapping of the entire surroundings, building a 360° point cloud with respect to the object it is mounted on (in our case the drone). This sensor is very expensive and has a non-negligible weight for a drone of this size (having in mind to reduce the weight as much as possible, this could be a problem), moreover it is very sensitive to vibrations, so it is very difficult and not very worthwhile to mount it on a drone that during the flight will never be perfectly steady therefore the resulting measurements would be not very precise and with a non-negligible error.

UWB - Ultra Wide Band:

This solution consists of a constellation of n emitter antennas (satellites), of which the absolute position is known (with high accuracy), and a receiver antenna mounted on the drone (which moves in space), thanks to which it is possible to compute the relative distance with respect to the n fixed antennas. Consequently, knowing the relative distances of the receiver antenna (drone) with respect to the n fixed antennas, using various techniques similar to phone cell triangulation, it is possible to calculate the position in space of the receiver antenna (and thus of the drone) with reasonably high accuracy. The antennas should be mounted at the edge of the working area, as the accuracy is much higher when the object to be located is within the area enclosed by the antennas, otherwise, albeit with low accuracy, they are able to locate the receiving antenna even if it is located outside the area delimited by them.

To get information about the 2D location (X and Y) of the drone you need n+1 antennas: where n is the number of dimensions to be identified. So if we want to have a 3D location in space (X, Y and Z) we need at least 4 antennas, it should be noted that to have a location in space, the antennas must be mounted on different planes, otherwise if they were on a single plane it would not be possible to get the location in space because the intersection of the spheres would be such that it would not give the information of the 3 dimensions but only 2. From the studies carried out, it has been noted that the more the number of antennas, the lower the error will be, thus the higher the accuracy of drone localization.

For some applications, it is also possible to think of using the UWB localization system for localization in the X-Y plane (with the number of antennas correctly sized according to the typology and extension of the area in which the drone will have to operate) and an infrared (or ultrasonic) sensor for the Z position, so as to calculate the height above the ground independently and very precisely. In this way, by making the localization in the X-Y plane independent of the localization in the Z height plane, a very high accuracy in spatial localization can be achieved.

It must be said, however, that the application of UWB technology in a real warehouse environment requires careful consideration of physical obstacles and the need for antenna power. These factors could limit the effectiveness, in fact, obstacles drastically reduce the accuracy of this system. While at the level of feasibility, it must be considered that these antennas must still be installed inside the warehouse: this operation, besides being very costly and impacting warehouse operations, is not definitive anyway, as the antennas must still be periodically inspected and maintained, to make sure they are working properly.

Implication of UWB in a real warehouse environment: Thinking about the real environment of a warehouse, however, other considerations have to be taken into account regarding the interference and/or reflections of the signals with the physical structures that stand between the emitter and the receiver (non-LOS reading - non Line Of Sight). In the open area, an accuracy of the order of about 10 cm is reported with 4 antennas at a distance of approximately a hundred meters; while it has been verified through empirical studies that the error grows much larger as soon as an obstacle is placed between the emitting and receiving antennas. In addition, we have to consider the fact that in a real warehouse context, these antennas would have to be powered anyway, and it is unfeasible to wire an entire warehouse to power such a system or to have a monthly or weekly replacement of batteries. For this reason, therefore, a localization solution based on UWB antennas, in a warehouse context in which there are the metal structures of the racks together with the pallets and goods that create a physical obstacle to signal propagation, in addition to the power supply problem, is probably not the most suitable technology for this scope.

VIO - Visual Inertial Odometry:

Is a technique, that using cameras and sensors of different types, through image processing, gyroscopes, accelerometers, and inertia sensors, thanks to complicated algorithms, is actually able to build a virtual map of the environment around it and thus calculate the relative position (with respect to an origin) and attitude of the drone (on which they are mounted). In particular, a relatively small drone that is supposed to navigate with agility within the aisle of a warehouse can be equipped with the following sensors:

- *Intel T265:* is a stereoscopic camera that uses data from optical, acceleration and inertia sensors, through a Visual Processing Unit is then capable of providing output of its position and attitude, along with all the data needed to locate the drone in space. This data will then be used by the Jetson Xavier Nano to decide what to do to proceed with the mission.
- *Intel D455 o in alternativa D435:* is another optical sensor equipped with stereoscopic, infrared and depth camera; with this data, it is possible to use a point cloud to map information about the surrounding environment such as obstacles, etc. in this way, it is possible to real-time map the environment and avoid any unexpected obstacles.

Thanks to this equipment, the drone is able to navigate autonomously within an environment such as a warehouse. It is also important to mention that these visual systems, work on recognizing some specific features like edges, boxes square rectangles, etc., in order to extract as much information from the environment, hence an environment like a warehouse is very suitable as it is rich in features and details (unlike for example a white room or an outdoor environment) that can be

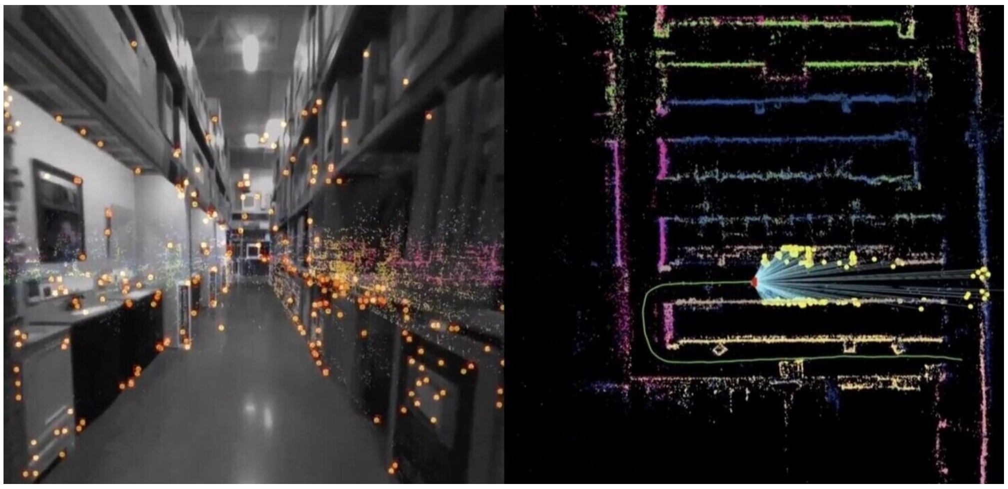


Figure 2.5: VIO (Visual Inertial Odometry) example

detected and extracted by the visual systems described above. Obviously, these systems in order to work need the environment to be properly illuminated, with no reflections or strange shadows that could undermine this system. it should also be noted that installing a light source on board the drone, besides being very energy consuming, could create serious problems in computing the position of the drone as the shadows that would be generated would not be solid with the space reference system, but would change in shape and position based on the light source (the drone) which, however, is moving. These shadows are in fact features that could be used by the VIO algorithm to determine position.

Fiducial marker

QR codes or other types of markers (easier to recognize like fiducial markers) can be used to provide relative referencing of an object's location in space. The environment in which the drone should move is prepared in advance by placing the markers equispaced (on the floor and/or rack), in order to build a kind of "map" of the working area based on the markers. The drone has a simple RGB camera that it uses to identify markers.

Because it is aware of the mapping that relates each marker to a specific location in space, it can determine its location in space by analyzing the marker it is reading and how it is reading it (at what angle and at what distance). This allows it to determine how to move in order to continue with its navigation. As it moves, the drone will recognize new markers and correct its trajectory accordingly. The use of QR codes can thus provide a relative frame of reference for an object's position in space. However, this system requires preliminary preparation of the environment,

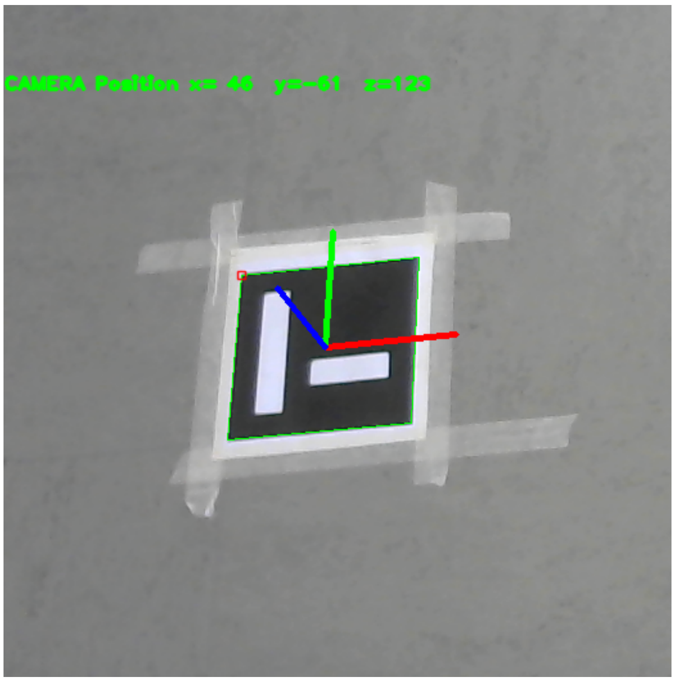


Figure 2.6: Fiducial Marker example

with the markers positioned equidistantly. While this could be a low-cost and relatively simple solution, it could also require significant time-effort to prepare the environment, and thus an economic loss from the warehouse performance.

2.6.3 Standalone drone and swarm of drones

As we can imagine a standalone drone's battery life will be limited, thus in order to do an inventory, it will need to go back to the base to recharge and pick up where it left off. However, carrying out things this way wastes a lot of time and notably consumes a lot of charge when moving from the charging base to the aisles that need to be inventoried. There are ways to spread out the recharging bases across the warehouse rather than having them all in one location. By doing this, the battery may be fully utilized for the inventory rather than being misused traveling hundreds of meters from the recharging base to the aisle. Finally, by expanding on this idea even further, it is possible to imagine approaching inventory in a different way, so that instead of one drone performing the inventory of a warehouse in sections, many drones (properly sized) would be in charge of performing simultaneously different parts of the warehouse (for example, only two or three aisles per drone). The benefit of this approach is definitely that by parallelizing the work, it is no longer necessary to wait for all of the drone charging cycles to finish the inventory and receive back the results, but more importantly, the system's complexity remains relatively constant. In fact, if the drones are autonomous, they will simply communicate data and their status to a central system; however, since all the complexity of navigation, etc. is onboard the drone, this approach would be very scalable. Since there is no interaction between the drones because they operate in separate and distinct areas, the complexity of the onboard system would remain unchanged. Therefore a central control and coordination system would be added that assigns the various missions to the different drones. With this strategy, the battery issue is slightly mitigated because we can quickly complete the inventory by using many drones.

Final consideration on standalone Drone solution

In order to have a robust and stable solution usually it is used a redundant system, i.e. you do the same thing but in different ways: in our case, you might think of having a navigation system that uses Visual Inertial Odometry at the same time alongside marker-based localization so that if one of the 2 systems fails, the other is still able to land the drone safely or alternatively get it back to its base. The solution that could be experimented with in the lab could be set up with 2 Intel cameras: T265 and D455 mounted on the face of the drone to allow navigation and obstacle avoidance. A simple little camera pointing downward for navigation based on markers placed on the floor, as well to help stabilize the hovering flight (possibly complemented by an infrared or ultrasonic sensor to measure height), and finally another simple camera that will read either any additional markers on the rack useful for navigation (and correction of drift errors, etc.), as well as codes on pallets to do inventory. In all of these cases, the battery life that can be achieved is about 20 minutes of flight time since all of the sensors and processor on board are energy consuming and drain quite a lot of power (in addition to the drone flight).

2.6.4 Drone supported by an AMR

For the sake of clarity, let us briefly explain the difference between AMR and AGV.

- *AGV (Automated Guided Vehicle):* A robot that follows fixed paths, typically requiring infrastructure changes like magnetic tapes, wires and sensors. These routes are predefined and require extensive installation, which can be costly and disruptive to production. AGVs have minimal onboard intelligence and can only follow simple programming instructions. They can detect obstacles but cannot navigate around them, so they stop until the obstacle is removed.

- *AMR (Autonomous Mobile Robot):* A robot that operates autonomously and can navigate in an uncontrolled environment without the need for fixed paths or tracks. AMRs are known for their flexibility, they are equipped with intelligent navigation capabilities thanks to the usage of advanced cameras, sensors, and laser scanners along with sophisticated software to construct maps and navigate autonomously. Unlike AGVs, AMRs do not rely on fixed routes. They can detect obstacles and safely maneuver around them by choosing the best alternative route.

Introduction

Other solutions include the usage of a drone supported by an AMR, again there can be different set-ups: the main difference between the possible set-ups is to have the drone connected to the AMR by means of a cable, from which it receives power or a free drone that uses the AMR only as a support to locate itself. For both setups, the basic idea is that the drone, while flying, has a lower localization accuracy than an AMR that is on the ground; therefore, the advantage of this solution is to exploit the ground robot's positioning accuracy to correct the positioning of the drone and keep it perfectly centered on the AMR's perpendicular (we will see later how).

In this case, the computation of positioning and the movements to be made are done on board the AMR, and the drone will simply follow its movements and commands (the intelligence is on the robot). In this type of system, localization of the system is simplified: basically, it is like reducing the complexity of the localization problem from 3 dimensions to 2 dimensions: X and Y. The third dimension (the Z) is independent since the AMR is a ground system and can be computed separately with other sensors, thus simplifying the navigation in X and Y of the AMR.

Since this use case requires a very high level of accuracy, we would employ the ground robot's localization capabilities to enhance localization. Additionally, by utilizing a system in which the AMR also powers the drone, we would be able to solve the battery issue and acquire a highly autonomous system that is also energy-efficient and has a long operating range. Last but not least, there would be certain benefits even at the level of safety: in fact, an AMR in this type of environment is undoubtedly visible and would prevent operators from passing underneath any drone that is flying overhead.

The used technologies

Wheel odometry, visual odometry, and Lidar are some of the various techniques that the AMR can use to identify its location and determine its orientation in space. In this case, since it is a ground system, the surrounding environment in which the AMR will have to operate can be easily mapped: for example, by manually moving it within the area, it is possible to create a relationship between the physical space and the values read by the wheel encoders, this information later will be useful together with the vision-based systems for orientation, localization, obstacle detection and navigation.

Visual Inertial Odometry Similarly to how it is done on the drone, through optical sensors such as the Intel T265, visual odometry can be exploited to get information about the surrounding space and thus track the movement in space of the AMR system. Again, here the more feature-rich the surrounding environment, the better the algorithms for feature extraction will work.

Lidar Since it is aboard a terrestrial robot, it is also possible to mount additional sensors such as a 3D lidar (e.g. the Intel L515 sensor): indeed, in this case, you do not have the problem of minimizing the weight (as on the drone) and you have significantly fewer vibrations, so it is conceivable to also use this technology to map the surrounding environment and thus also have an additional and very reliable obstacle avoidance system.

Wheel Odometry Using encoders mounted on the wheels of the AMR, it is possible to calculate how many revolutions the wheels completed, so by knowing the radius of the wheel and considering the wheel slippage on the floor negligible, it is possible to calculate the path taken by the robot and thus its relative position with respect to the starting point. Obviously, this relative position is subject to drift and therefore needs to be reset periodically, for example, using fiducial markers

Drone - AMR: mutual tracking Mutual tracking between drone and AMR is the main aspect of this solution; this can be done in 2 ways. The first one is to mount a camera on the drone that points downward, and by tracking a marker positioned above the AMR, based on how it is "seen" by the camera mounted on the drone (how it is distorted and its size), it can easily determine the relative position of the drone with respect to the AMR and correct accordingly, see figure 2.7. Alternatively, the camera can be mounted on the ground robot pointing upwards to track the drone by means of colored LEDs mounted on the booms of the drone, in this way the AMR that knows where it is in space "pilots" the drone sending commands to keep it centered on its perpendicular. In figure 2.9 you can see an example of this tracking light system.

Additional computing capacity A compute unit like the NUC, which offers more computing power and better flexibility in installing development packages, can be mounted on board the robot because space and weight constraints, which are no longer relevant in this case, make it possible. This unit can then be flanked by

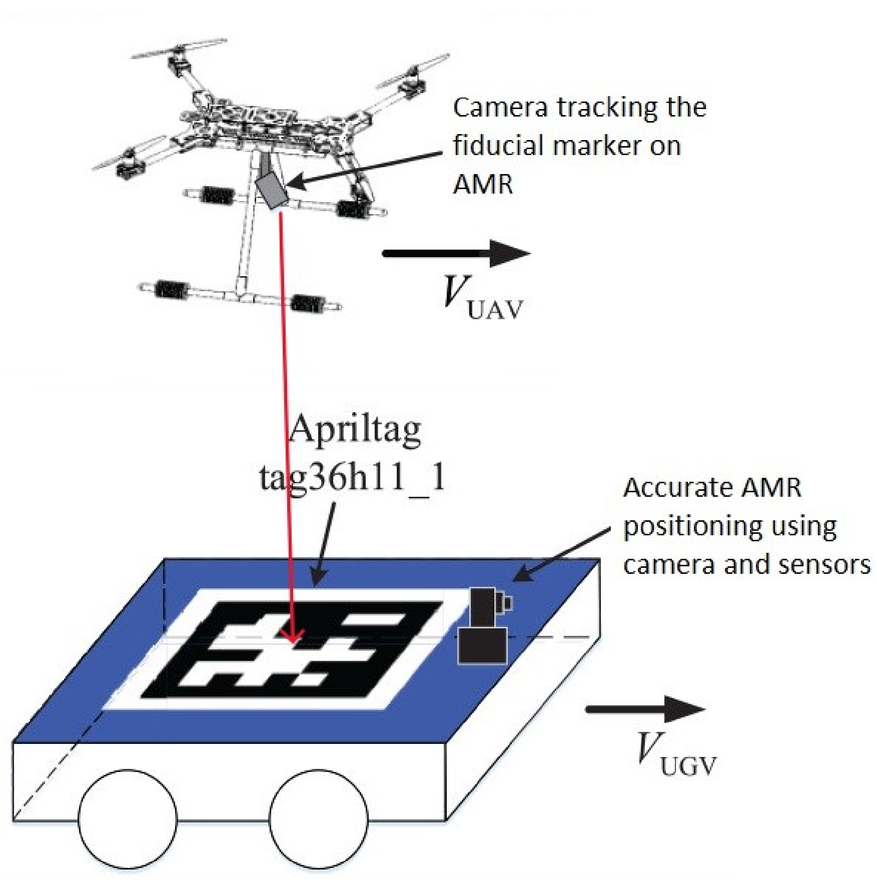

Camera tracking the fiducial marker on AMR
V_{UAV}
Apriltag tag36h11_1
Accurate AMR positioning using camera and sensors
V_{UGV}

Figure 2.7: The drone tracks the fiducial marker on the AMR, to correct its position and trajectory

a Coral, which is essentially an external GPU from Google, allowing for expanding computation to support neural networks and other technologies.

Omnidirectional wheel The cost of the AMR that can be used can range from a few hundred euros to several tens of thousands of euros, so this must also be taken into account when analyzing the solution's viability.

AMRs with omnidirectional wheels maintain the reference system's orientation because there is no rotation of the YAW angle, as opposed to AMRs with standard wheels that move in space and change the YAW angle (i.e., rotation around the Z axis). With these special omnidirectional wheels (2.8), in fact, an AMR is able to move along X and Y but keeping its YAW angle fixed, this greatly facilitates and eases the computation and algorithms for navigation as in fact one degree of freedom is eliminated; on the other hand, the complexity falls on controlling the drivers of these particular wheels which will also be somewhat more expensive than

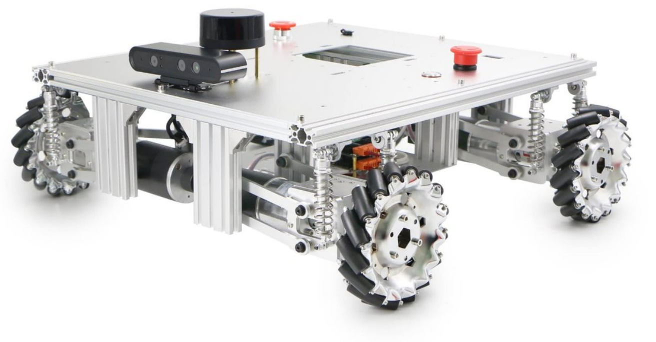


Figure 2.8: Example of an omnidirectional wheel on an AMR

traditional ones (and probably more fragile on a floor like a warehouse one). It's crucial to remember that the price of an AMR can vary widely, from a few thousand euros to several tens of thousands. Of course, the payload, sensors, and other components mounted on board, as well as the typology and construction of the robot (i.e. if has industrial standards or not), have a significant impact on the price. For instance, AMRs with these specific wheels fall into a different pricing band, and the cost quickly soars. Omnidirectional wheels are actually highly expensive due to their sophisticated design and the development of their complex drivers. Additionally, because of their unique design and functionality, they can only be used on surfaces that are extremely uniform and free from flaws.

Cable power supply Finally, a key difference, as anticipated, is having the drone powered by means of a cable, or completely free. With no weight restrictions (unlike on a drone) in the first scenario, it is possible to have a large battery on board the AMR, which allows for a battery life of several hours (up to six). However, this solution has the drawback of adding a cable tensioner (winch) as a new part of the system (2.9). The cable will need to automatically roll and unroll every time the drone changes altitude, and while this operation may appear to be very straightforward, there are many factors that could go wrong and cause a crash. For instance, if the wire is not rolled correctly for some reason and gets caught in the drone's propellers, or if it is not unrolled quickly enough, it could cause instability in the drone's flight and crash. The wire needs to be slightly loose so that the drone can fly freely and correct its position, but not excessively tight so that it runs the risk of getting tangled in the propellers during the drop phase.

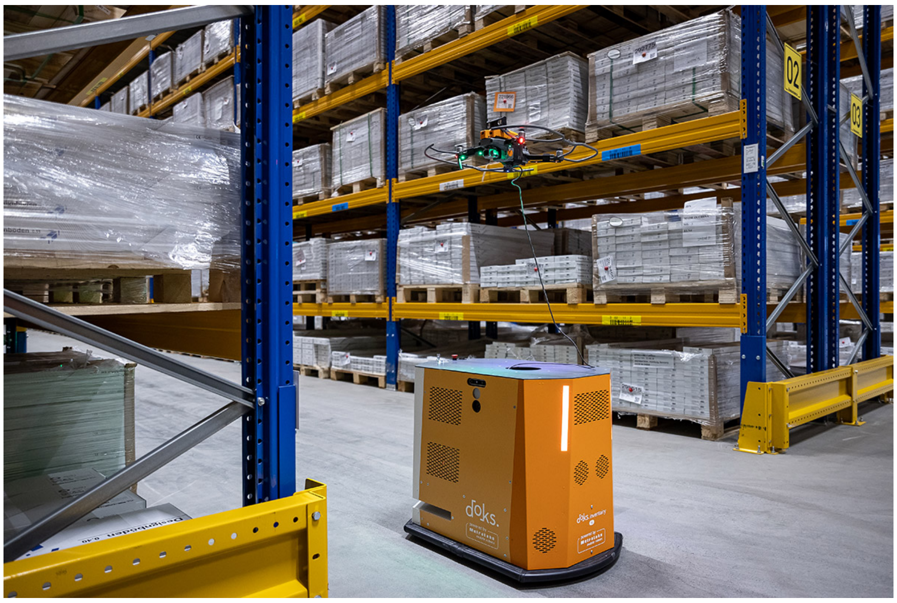


Figure 2.9: AMR that supplies power to a drone with a cable

LED light system With such a large battery, another option for this kind of solution is to mount a lighting system on the drone and AMR. This is actually a great advantage as it is possible to take pictures and carry out inventory in reduced brightness or even no light conditions. This means being able to carry out inventory at night with total autonomy and the absence of personnel saving a lot of money on lighting, which can then be turned off.

Standard (battery-powered) cordless drone power supply The alternative solution, on the other hand, loses the major benefit of having a longer autonomy because the system's autonomy would be constrained by the drone's autonomy, which is approximately 20 minutes. This alternative solution is simpler from a technical point of view. In this case, therefore, the advantage of having an AMR helping the drone to position itself would not be compensated by the additional cost and complexity of having a drone and an AMR.

2.6.5 AMR with multiple cameras mounted on a pole

The third option is an AMR that has a pole mounted on it with multiple cameras installed upon it. Again, there are two versions, one that uses a rigid and fixed pole and the other that uses a telescoping pole. The AMR uses the exact same technologies for navigation, but in this case, there is a pole with a camera that takes pictures of the higher locations instead of a drone. Since the AMR can be set up the same way as in the previous case, the navigation of the AMR is carried out using the same methods and techniques.

AMR with fixed pole

The version with the fixed pole is really basic since, in addition to the AMR, it only has a simple pole mounted on top as a payload that has a camera for each shelf so that it can take pictures and perform inventory. In fact, if the pole were very high, the system might have issues moving between shelves or from one area of a warehouse to another because of the significant fixed encumbrances that prevent this solution from reaching very high shelves. This is a very constraining and limiting requirement that would also require the warehouse to be free of overhead impediments, even outside the aisles.

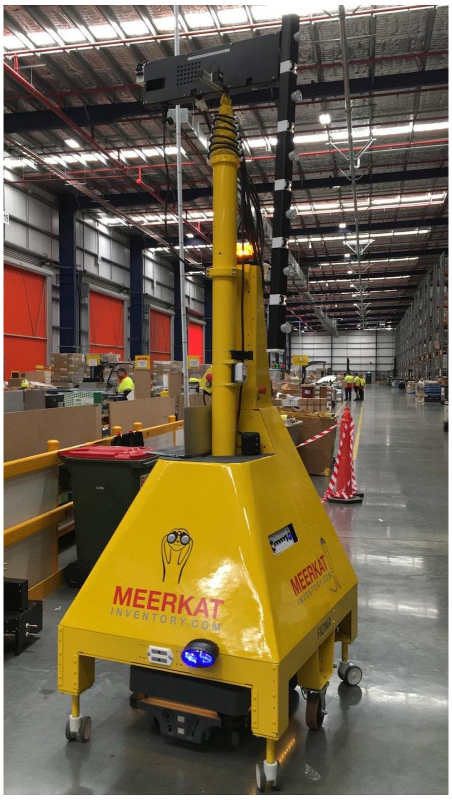


Figure 2.10: Example of AMR with a telescopic pole and a camera

AMR with telescopic pole

The version with a telescopic pole partially solves the space problem in that the pole once it is retracted takes up very little space and the system can also fit under a very low gate, however, the complexity of the solution increases considerably, because engineering a telescopic pole with a camera on top that needs to be mounted on an AMR is not simple. It must be added that in this case as the pole rises, it is subject to a lot of vibration because, given the height of the pole, there would be oscillations due to the movement of the AMR.

Consideration on AMR with pole

To conclude although this solution seems apparently simpler than the previous ones, it is not widely used because of the various problems we have mentioned, which limit its effectiveness and functionality in real inventory operations.

2.6.6 Final consideration on these solutions: pro and cons

Taking into consideration what has been said therefore, the 2 most prospective solutions and the ones that can actually be used are the stand-alone drone and the drone supported by AMR, the one with the pole is to be excluded because of the numerous issues mentioned.

Standalone drone

The stand-alone drone uses markers and visual odometry to orient and navigate, and it can take inventory using one or two cameras mounted on the sides to read the QR codes on the pallets stored in the rack. The primary drawback of this solution is the flight time autonomy, which is definitely time-limited, in fact, it is around half an hour of flight time. However, this solution has the advantage of being very scalable because each individual drone is inexpensive (compared to the other solution), so it can be designed to scale up the solution based on the size of the warehouse in which the system will have to operate. In this case, in fact, by expanding the operational fleet of drones, the complexity of the system does not increase because each individual drone (having all the navigation logic and "intelligence" on board), is able to navigate and carry out its intended function. Possible developments could lead to the development of an autonomous charging system, such that when the drone returns and lands on its charging base, it automatically recharges its batteries, thus solving the problem of manually changing batteries.

Drone supported by an AMR

Usage of the drone supported by the AMR in a configuration where it is powered by a large battery mounted on the AMR would constitute a workable alternative. In this way, it is true that the winch and the mechanism that rolls and unrolls the cable increase the complexity of the system, but this increase in complexity would be countered by the system's considerable amount of autonomy, allowing it to run for several hours (up to 6 hours) without stopping or requiring recharging. The cable-less version, on the other hand, would not justify the additional complexity and cost, and would also be less scalable as the cost per unit of AMR plus drones would still be higher and costly for this scenario, making it impossible to have a fleet.

Chapter 3

State of the art for indoor navigation

3.1 What is SLAM?

Even though there are many standalone mapping and localization methods available, SLAM's complexity arises from combining mapping and localization at the same time. For a long time, it seemed that having a robot creating a map while also keeping track of its own location amounted to a classic "chicken or the egg" dilemma with no obvious answer. However, a variety of approximations have come close to resolving this challenging algorithmic problem after decades of mathematical and computing effort. The main challenge in the fields of mobile robotics and artificial intelligence is simultaneous localization and mapping (SLAM), which deals with the issue of localization and mapping when a previous map of the workspace is not available.

Localization An autonomous robot's initial action after being turned on is to locate itself. A robot is typically equipped with sensors to scan its surroundings and keep track of its motions in a localization scenario. The robot can locate its location on a given map by using the sensor inputs. A tracking device, such as a GPS, may occasionally be utilized to support in localization

Mapping Even though it might seem straightforward, creating a map is a difficult task, especially for a robot. To start with, a visual sensor of some kind, such as a camera or a lidar sensor, is used to capture the surroundings. As the robot goes, it collects more visual data, tries to connect the dots, and extracts features to indicate certain distinguishable landmarks, like a corner. However, some of the traits could be very similar, making it challenging for a robot to distinguish one from another. At this point, localization can be used with SLAM to produce more precise maps.

Landmarks Landmarks are features that are simple to re-observe and differentiate from their surroundings. The robot uses these to locate itself and determine its location. Imagine yourself with your eyes covered to get a sense of how it works for the robot. In order to avoid becoming lost while moving around in a house while wearing blinders, you might reach out and touch things or bump into walls. You can get an idea of where you are by noticing distinctive details, such as the surface of a doorframe. Robots can feel touch thanks to sonar and laser scanning technology. Landmarks should be re-observable by, for instance, being able to be seen (detected) from different points of view and angles.

Landmarks need to be unique to be recognized from one time-step to the next without being confused. In other words, if you subsequently re-observe two landmarks it should be simple to identify which one is which of the landmarks we have already observed. This could be challenging if two landmarks are similar to each other. The number of landmarks present in the environment and that the robot is able to recognize and select must not be too low, otherwise too much time would elapse between the observation of one landmark and the next, which could lead to non-negligible errors in SLAM and cause the robot to lose

To summarize the key points about suitable landmarks are:

- Landmarks should be easily re-observable
- Each landmark should be unique and recognizable from the others
- There should be several landmarks around in the working environment
- Landmarks should be stationary

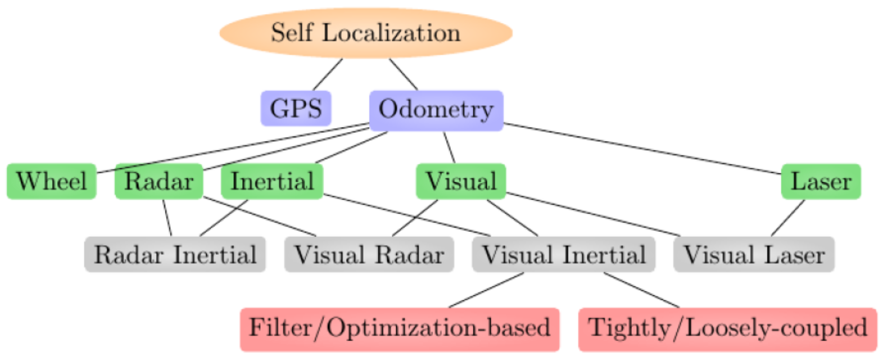


Figure 3.1: Categorization of the localization techniques proposed in literature

Odometry Data The purpose of the odometry data is to give an approximation of the robot's position as determined by the movement of its wheels, which can then be used to make a first guess as to where the robot might be in the EKF. Getting the synchronization correct with the laser and odometry data is challenging. If the odometry data is retrieved later, the laser data at time t will be out of date. It is possible to extrapolate the data to ensure that they are valid simultaneously. Since the controls are known, extrapolating the odometry data is easier. The results of the laser scanner measurements can be quite difficult to predict. It is simplest to request both the laser scanner values and the odometry data at the same time if there is control over when the measurements are returned.

3.1.1 Challenges with SLAM

Sequential movement is estimated by SLAM and has some error tolerance. Over time, the error accumulates leading to in an important deviation from the true values. Additionally, it may result in map data collapsing or distortion, making following searches tricky. Consider for example navigating a square-shaped path, robot's starting and final points get mismatched as the errors accumulate: this is the loop closure problem. Another scenario would be when the environment where SLAM is performed is dynamic, so it would needs to updates. Imagine a robot strolling down a warehouse corridor that is empty, then on the way back it finds pallets on his path. Would it be able to complete the loop and realize that it is still in the same location but with new things added? These kind of pose estimate errors are inevitable. It is important to detect loop closures and determine how to correct or reset the accumulated error. A possible countermeasure is to use landmarks from previously visited places to reduce the localization inaccuracy. To aid with error correction, pose graphs are created. Error minimization can be solved as an optimization problem to produce more precise map data; this type of optimization is known as bundle adjustment.

Image and point-cloud mapping does not take a robot's movement characteristics into account. This method may occasionally result in discontinuous position estimates. An analysis report, for instance, showed that a robot moving at 1 m/s suddenly leaped forward by 10 meters. This type of localization failure can be avoided by combining the motion model with multiple sensors in order to perform calculations based on the sensor data, or by utilizing a recovery algorithm.

The use of a motion model with sensor fusion can be done in different ways. Kalman filtering is a prevalent technique for localization. Extended Kalman filters and particle filters (Monte Carlo localization) are frequently utilized because nonlinear motion models are typically used by differential drive robots and four-wheeled vehicles. In some circumstances, flexible Bayes filters, like unscented Kalman filters, can also be used. Inertial measurement units (IMU), Attitude and

Heading Reference Systems (AHRS), accelerometer sensors, gyro sensors, odometry (using enconders attached to the wheels), and magnetic sensors are some examples of most commonly used sensors. When localization is unsuccessful, a recovery strategy is to recall a landmark as a key-frame from a previous location. A feature extraction method is used when looking for a landmark so that it can scan quickly. Some techniques based on image features like bag of features (BoF) and bag of visual words (BoVW) can be used. Also deep learning has been used for comparing distances from features.

Implementing SLAM on a vehicle's hardware presents a difficulty with computing costs. Embedded microprocessors that are small and low-energy typically have a limited processing power. It is crucial to process image and do the point cloud matching with high frequency in order to get accurate localization. All optimization computation as for example loop closure are very high computation processes: the challenge is to carry out such computationally complex computations on embedded microcomputers.

For this reason, depending on the final application it would be necessary to create just the relative position of obstacles (topological method) or the exact distances between them (metric method). Furthermore, it is possible to choose if recreate a digital copy of the park (volumetric method) or just enough information to distinguish objects (feature-based method). Depending on these parameters the computational effort vary a lot.

One countermeasure could be to run different processes in parallel. The parallelization of processes like feature extraction, which is a preprocessing step in the matching process, is often feasible. Speeds can be increased in some circumstances by using embedded GPUs, SIMD (single instruction multiple data) calculations, and multicore CPUs for processing. Additionally, as pose graph optimization can be executed over a long cycle, lowering its priority and carrying out this process at regular intervals can also improve performance.

Last but not least is the multi-robot case. When there are multiple autonomous robots navigating inside a warehouse, might occur different challenges. One challenge is the ability to create a clean map without other robots. How can you be sure that every robot is aware of the locations of the others so that you can exclude them from mapping? Can each robot construct a smaller map locally and share it with a central system to create a whole map if they can interact with each other?

3.2 Camera-based SLAM

The employment of simple, low-cost cameras (wide angle, fish-eye, and spherical cameras), compound eye cameras (stereo and multi cameras), and RGB-D cameras (depth and ToF cameras) enables the recording of high-resolution images with rich details. In general, there are two types of visual SLAM algorithms. Sparse methods use algorithms like PTAM (Parallel Tracking and Mapping - for augmented reality), ORB (Oriented FAST and Rotated BRIEF), and SIFT (scale-invariant feature transform) to match feature points in images. Dense approaches make use of algorithms like DTAM (Dense Tracking and Mapping), LSD (Large-Scale Direct Monocular), DSO (Direct Sparse Odometry), and SVO (Semi Direct Visual Odometry) as well as the overall brightness of the images.

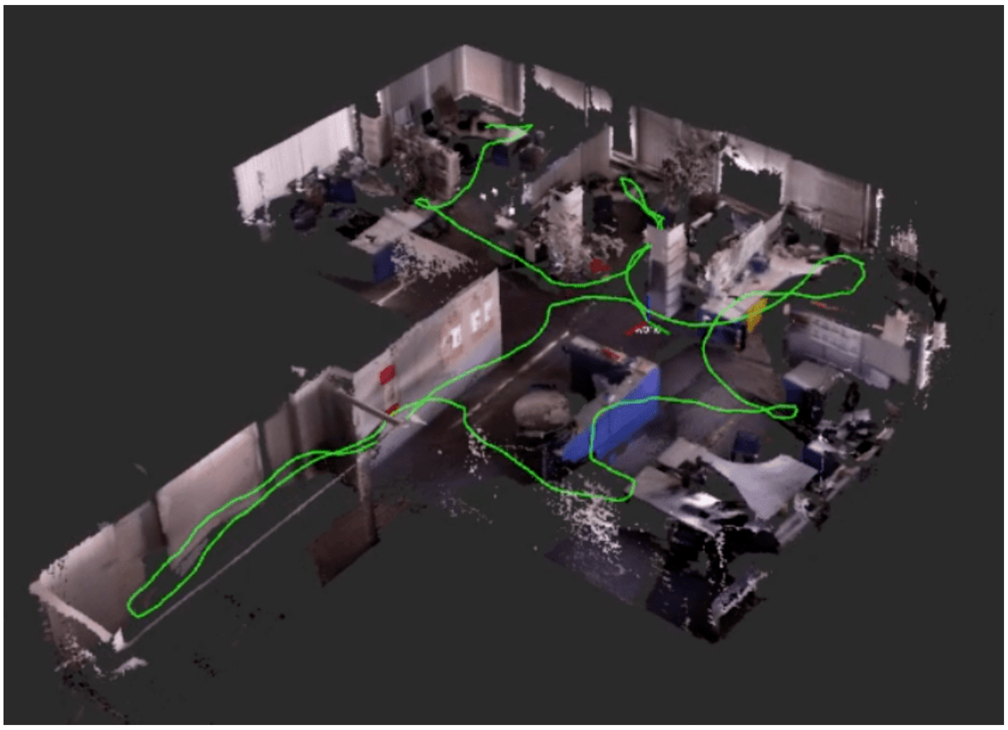


Figure 3.2: Example of a Visual SLAM

For instance, the images that are captured when a robot uses a camera to map a warehouse may include pallets, shelves, and doors. The robot can determine that there are several items in the scene by comparing the color differences between adjacent pixels 3.2. Additionally, because cameras produce a lot of information, they can be utilized to find landmarks (already determined positions). Additionally, landmark detection and graph-based optimization can be used to achieve flexibility in SLAM implementation. It is difficult to quantify depth with monocular SLAM, which uses just one single camera. This limitation can be resolved in one of two ways: either by locating AR and fiducial markers, checkerboards, or other well-known items in the image for localization, or by fusing camera data with data from other sensors like inertial measurement units (IMUs), which can measure velocity and orientation. Structure from motion (SfM), visual odometry, and bundle adjustment are examples of vSLAM-related technology.

3.2.1 VO

Visual odometry (VO) is a technique that estimates the position and orientation of a platform by analyzing variations in camera motion on a sequence of images. It is part of Structure from Motion (SfM), a technique for reconstructing a 3D scene and camera pose using images. SfM can be performed based on camera number, calibration status, and image order. A 3D scene is reconstructed by computing the OF (optical flow) from key information extracted from two consecutive image frames. The key information (e.g. corners) is extracted using an image feature detector, such as Moravec [15] and Harris [16] corner detectors. The reconstructed scene can be improved through bundle adjustment or offline optimization methods. Three standard techniques are used to calculate the transformation matrix between sequential images based on point correspondence specifications in two or three dimensions.

- *3D-3D correspondences:* Camera motion can be computed using two sets of three-dimensional correspondences. First, two stereo image pairs are captured, feature points are extracted and matched, then the 3D matched points are triangulated. The transformation is computed with an absolute scale by minimizing the L2 distance between the two points.
- *2D-2D correspondences:* The transformation matrix is calculated using the essential matrix, which defines the geometric relationship between two sequential images. It is computed from 2D feature correspondences using the epipolar constraint. The Nister five-point algorithm is a common approach, using five corresponding points to determine the relative scale between consecutive frames.
- *3D-2D correspondences:* The main concept of this method is to compute the transformation matrix by minimizing the 2D reprojection error from 2D and 3D correspondences [17]:

  $$T_t^k = \arg\min_{T_t^k} \sum_i |p_k^i - P_{t-1}^i|^2$$

  where T_{t-1}^t is the transformation matrix from $t-1$ to t, the image measurements are denoted as p^t, and P_{t-1}^i is the reprojection of the 3D features X_{t-1}^iinto image I^t. The perspective-n-points (PnP) problem estimates camera pose using N 3D points. [17]

The minimal solution requires three 3D-2D correspondences, known as the perspective-3-point (P3P). 3D-2D motion estimation offers better accuracy than 3D-3D due to minimizing image reprojection error instead of 3D-3D feature position error.

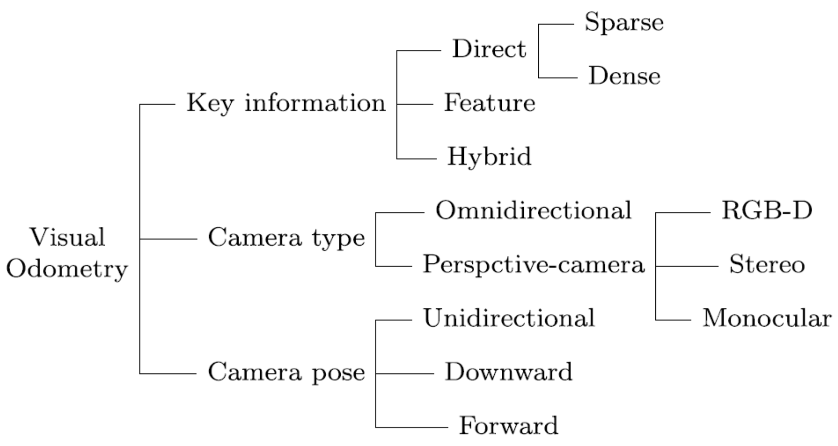


Figure 3.3: General classification of studies in the field of visual odometry.

VO techniques are categorized based on key information, camera position, and camera type/number. Key information can be direct raw measurements like pixels or indirect image features like corners and edges. Camera type/number can be monocular, stereo, RGB-D, omnidirectional, fisheye, or event-based. Camera pose can be forward-facing, downward-facing, or hybrid. More details on these VO techniques are provided in the following section.

VO-Approaches

1. **Direct approaches:** Direct approaches estimate vehicle position using raw visual measurements in pixels. The intensity of image pixels is analyzed to estimate the pose. The algorithm uses consecutive images from cameras to determine changes among frames using an optical flow (OF) algorithm. OF algorithms are classified into dense and sparse schemes. Dense OF optimizes all pixels based on global smoothness, while sparse OFs process some pixels from the whole image by solving the brightness constancy equation using a template matching technique. Both approaches use pixel intensity to compute the 2D displacement vector.

2. **Feature-based approaches:** Feature-based or indirect approaches are used to extract points of interest in images using feature detectors like corner or edge detectors. Corners are unique keypoints due to their two-dimensional intensity change, while edges are areas with strong intensity contrasts. Most edge detectors are based on gradient or Laplacian methods: the Laplacian detector uses one kernel to search for zero crossings in the second derivative of the image, while the Canny edge detector uses two kernels to identify

maximum and minimum in the first derivative that represent an edge. The feature-based or indirect approaches are used to extract points of interest in images using feature detectors like corner or edge detectors. Corners are unique key points due to their two-dimensional intensity change, while edges are areas with strong intensity contrasts.

3. **Hybrid approaches:** Feature-based approaches are less robust in low-texture environments due to their limited detection and tracking capabilities. Direct methods, on the other hand, exploit all key information in images, including weak intensity variations, leading to more robust and efficient results. However, direct methods are computationally more demanding. A hybrid approach combines direct and feature-based methods, mitigating issues and exploiting the benefit of the two methods. For instance, a semi-direct visual odometry (SVO) method has been proposed to eliminate the need for costly feature extraction at every frame, using subpixel feature correspondence to increase accuracy.

Camera type and number

- **Stereo:** Stereo camera setups use multiple cameras to reconstruct 3D information from stereo image pairs, allowing for accurate pose estimation. This is done by extracting and tracking key information between two pairs of images and applying a motion estimation algorithm. Epipolar geometry, is a technique that narrows the feature search domain from 2D images to 1D epipolar lines, reducing search time. However, stereo cameras require precise extrinsic calibration, which can degrade over time due to varying conditions like shocks and vibrations. Additionally, they typically have a fixed baseline distance, which affects depth estimation accuracy. Large baseline distances are needed for outdoor environments, but due to platform size limitations, it's difficult to have two cameras with a large baseline distance.

Figure 3.4: Example of a stereo camera with 2 optics distanced from each other

- **Monocular (standard camera):** Monocular setups estimate position and orientation by analyzing consecutive images from a single camera, which does not suffer from baseline issues like stereo cameras. Monocular visual orientation (VO) has gained attention in recent years but requires at least three different frames to reconstruct 3D information. One disadvantage is that the translation vector is computed up to a relative scale, as the transformation between the first two frames is not fully known. Solutions include obtaining additional information about the initial transformation using other sensors or relying solely on the visual information captured by the camera. An algorithm has been proposed to tackle the scale problem in translation using vision data, camera mounting point, and road surface planarity, in this way is possible to resolve ambiguity in scale and reduce scale drift. However, this method is infeasible for real applications due to the heavy computational needs of CNN (deep convolutional neural network).[17]

- **RGB-D:** RGB-D cameras offer a more efficient solution for obtaining real-depth information compared to stereo and monocular setups. Stereo cameras require costly epipolar line searches and additional warping processes, while monocular setups cannot provide depth information about surroundings on a real scale. Most RGB-D visual odometry approaches use feature-based methods for more robustness and direct-based RGB-D VO approaches are presented for accurate pose acquisition in low-texture environments and avoiding computing resource consumption in feature detection and matching processes.

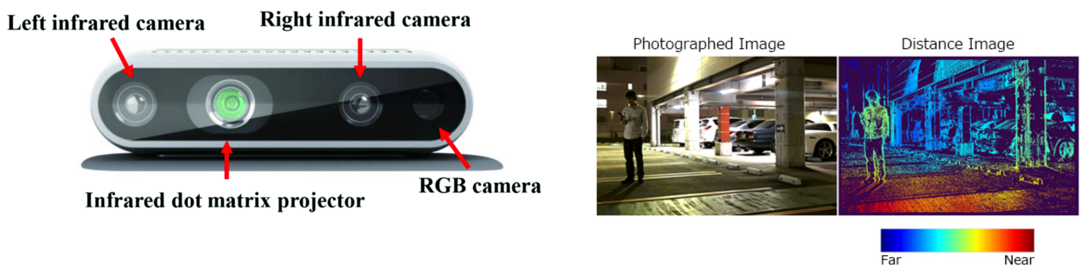


Figure 3.5: Example of an RGB-D camera and its output. Here we have the D435i which is the one used in our tests

- **Omnidirectional:** An omnidirectional camera (see fig 3.6), also known as a 360-camera, has a 360° field of view (FOV) in azimuth and 90° to 140° in elevation, and can include a fisheye lens or a catadioptric optical system (see fig 3.7). It can achieve more accurate pose estimation than traditional cameras with a small FOV and overcomes the rotation-translation ambiguity of small FOV cameras. An example of a VO system using a single omnidirectional camera is a vehicle orientation (VO) system, consisting of two modules: a homography-based module for detecting and tracking features from the ground plane and a direct module for estimating vehicle rotation.

Figure 3.6: Example of an omnidirectional, 360° camera

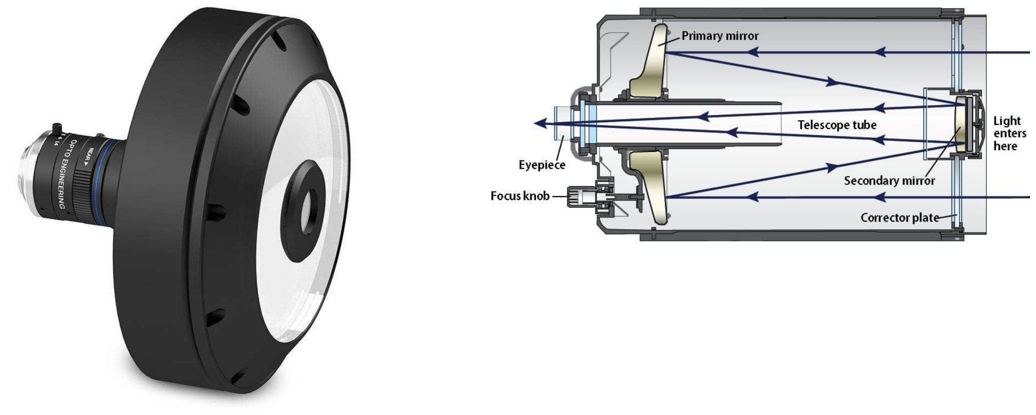


Figure 3.7: Example of a catadioptric camera and its working scheme.

- **Fisheye:** Fisheye cameras produce wide panoramic images with nearly 180° from side-to-side, allowing for more detailed observations of environments compared to pinhole cameras. However, they can produce distortions, necessitating a special distortion model and calibrations. Pixel correspondences in fisheye-stereo cameras are on epipolar curves, making traditional disparity search algorithms unsuitable. Additionally, these algorithms require more computational power due to the high cost of computing epipolar curves.

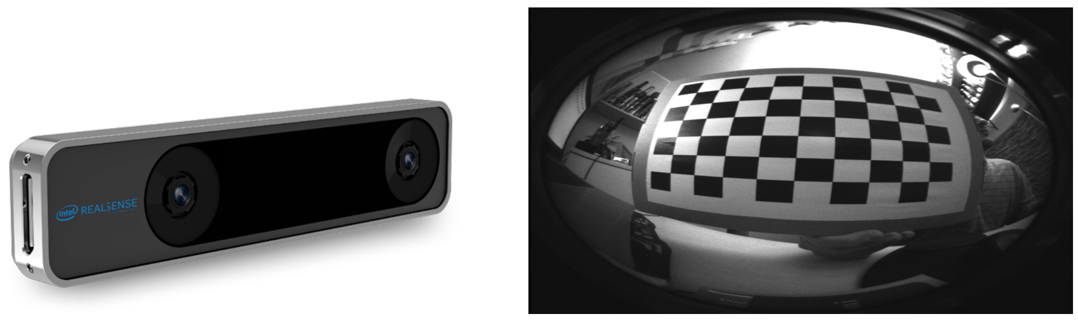


Figure 3.8: Example of a camera with fisheye lenses: T65, also used in our tests. The second picture is an example of type of picture it can take.

- **Event-based:** DVS, or dynamic vision sensors, are bio-inspired event cameras that capture changes in intensity asynchronously across all pixels on the camera. These cameras offer low latency, high temporal resolution, and high dynamic range (140 dB), compared to conventional cameras' 60 dB. They also do not suffer from motion blur, making them an improvement for vision-based localization algorithms like VO. For example, there is an algorithm to compute optical flow from the event stream that uses a feature-based method, building a polarization map for each pixel in the image. This map enables easy detection of motion by counting the number of incoming events with expected polarity.

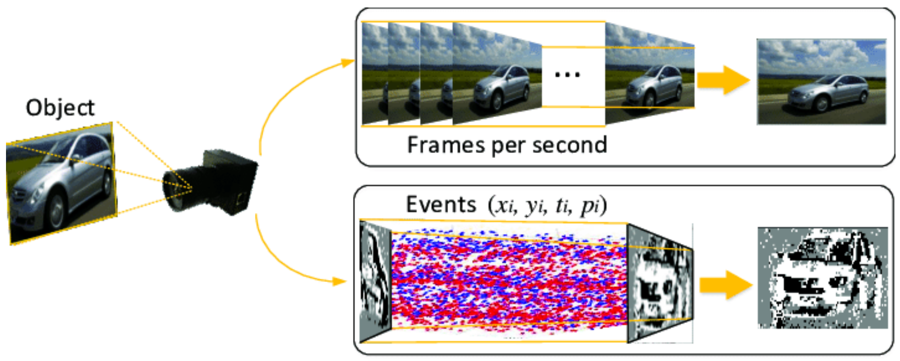


Figure 3.9: Event-based camera working scheme

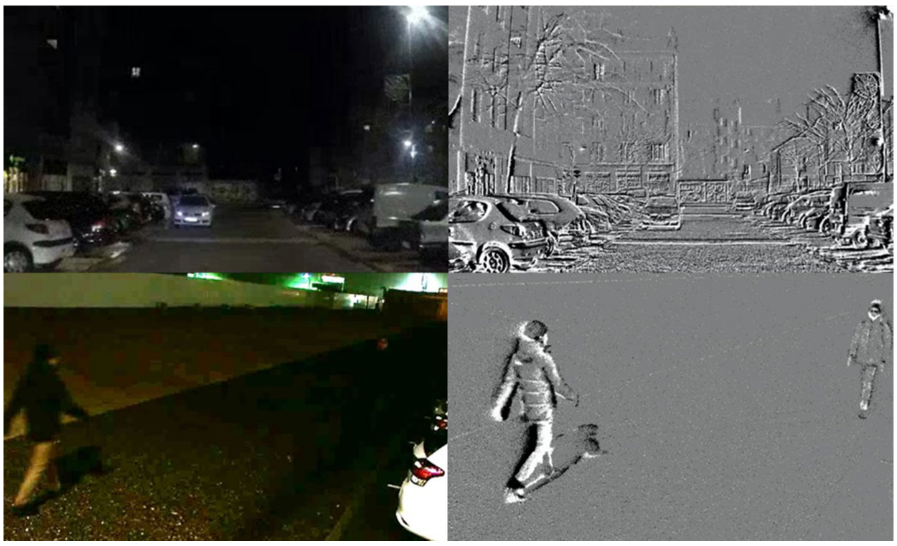


Figure 3.10: Event-based camera output example

Camera-pose

Visual odometry (VO) localization systems can be categorized into forward-facing, downward-facing, and hybrid setups. Forward-facing systems provide more information but are suboptimal for detecting small movements and can be obscured by shadows and surround changes. Downward-facing systems have been successful for positioning in pre-explored environments but are inaccurate when vehicles are moving fast. A hybrid approach combines both setups to solve these limitations.

Despite the success of visual odometry in indoor environments, there are some challenges to using VO as a precise localization method. Major challenges include computational complexity, scale ambiguity, and image conditions like lighting, low-textured regions, and blurriness. Additionally, VO suffers from drifting issues due to its incremental computation of the camera path, leading to the gradual accumulation of errors over time. To address these challenges, methods for sensor fusion, such as visual-laser and visual-inertial odometry, have been proposed. [17]

3.2.2 VIO

Localization methods based on vision are influenced by environmental conditions like lighting and shadows, while IMU-based methods deteriorate over time. Integrating these methods results in visual-inertial odometry (VIO), which offers greater accuracy and robustness. VIO can be categorized in different ways, considering how the visual and inertial data are fused there two categories: filter-based [3.2.2] and optimization-based [3.2.2]. Basing the categorization on when the measurements are fused there are three types of approaches: loosely-coupled [3.2.2], semi-tightly coupled [3.2.2] and tightly-coupled [3.2.2].

Loosely coupled approach Loosely-coupled techniques involve blending pose estimation from two standalone subsystems, a visual odometry module and an IMU module, to refine the pose of a robot. This approach limits computational complexity by using a fixed dimension for the state space. Common methods for fusing sensor data include the conventional Kalman filter (KF) and nonlinear optimization techniques, which offer better accuracy and robustness but require more computational power. Loosely-coupled approaches can be categorized into two main branches:

1. using pose data from the VO subsystem as the update step for IMU measurements
2. integrating data from the IMU sensor as independent measurements into a vision optimizer

The VO subsystem consists of two main units: the extraction and tracking unit and the VO estimator unit. Features are extracted and tracked from two consecutive images, and an ego-motion algorithm is applied to obtain position and orientation. The IMU subsystem estimates position and orientation by integrating measurements from the IMU sensor. The fusion is applied at the last stage of the pipeline to refine the position and orientation estimated by the two subsystems.

In [18], a loosely-coupled filtering framework is proposed to integrate noisy measurements from stereo cameras and an IMU sensor for more accurate and efficient real-time pose estimation for indoor-outdoor drones applications. The framework is based on an Unscented Kalman filter (UKF) instead of the popular Extended Kalman filter (EKF)-based framework, avoiding computational complexity and temporal drifting.[17]

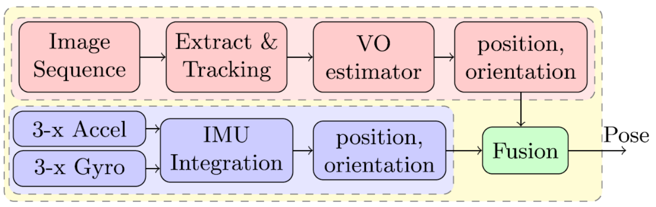


Figure 3.11: Block diagram of loosely coupled framework

Semi-tightly coupled approach Semi-tightly coupled approaches combine the estimated pose from the VO subsystem with raw measurements from the IMU sensor to achieve balanced accuracy and computational complexity. One example could be to use an optimization-based framework to fuse preintegrated IMU measurements with pose measurements from the VO module based on edge alignment. IMU pre-integration avoids determining global rotation between the world frame and body frame, which is challenging in many applications. Additionally, incremental rotation is initialized using the gyroscope reading to improve convergence during aggressive motion.

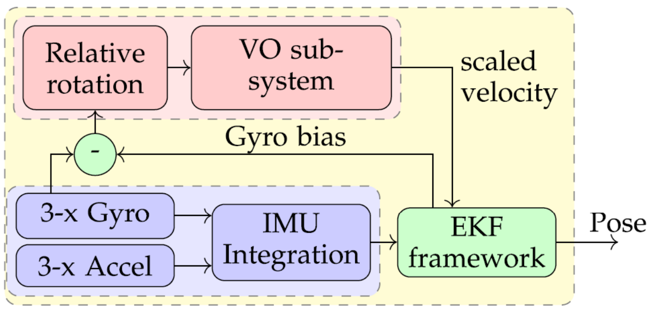


Figure 3.12: Block diagram of semi-tight coupling of inertial-optical flow.

Tightly-coupled approach Tightly-coupled approaches combine key information from captured images with raw measurements of the IMU sensor at early stages to achieve better accuracy. Key information are obtained using image detector techniques or pixel intensity with OF algorithms. This direct and systematic fusion of visual and IMU measurements leads to better results compared to loosely coupled approaches. Features extracted from captured images are fused with raw measurements from the IMU sensor to obtain more accurate pose estimation. Tightly-coupled approaches can be classified into filter-based and optimization-based.

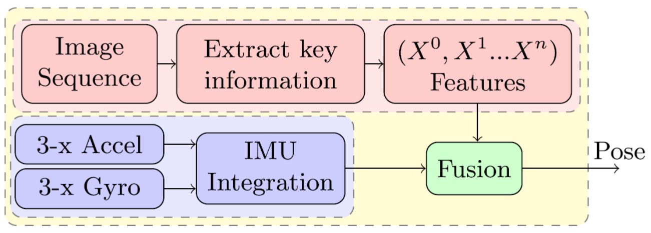


Figure 3.13: Block diagram of tightly coupled framework.

Filter-based approach Filter-based approaches are early methods used to solve VIO and SLAM problems. They consist of a prediction and update step, using measurements from internal state sensors like IMU sensors to compute the prior distribution of the platform pose and exteroceptive sensors like cameras to build the likelihood distribution. In filter-based visual-inertial odometry, the prior distribution of a vehicle is computed using linear and angular velocities from the IMU sensor. This dynamic model is used in the prediction step to predict the robot's motion. Key information from captured images is used as a likelihood distribution to update predictions in the update step. Filter-based visual-inertial odometry can be classified into three categories: extended Kalman filter (EKF) [3.4.1], multi-state constraint Kalman filter (MSCKF) [3.4.3], and unscented Kalman filter (UKF) [3.4.3]. The Kalman filter will be investigated in detail later in another section. [3.4]

Optimization-based approach Optimization-based approaches, to estimate the pose, use nonlinear optimization to minimize errors between integrated motion from inertial measurements from an IMU sensor and camera motion estimated through classic reprojection error minimization. These techniques outperform filter-based approaches in terms of accuracy, but require more computational resources. They can be divided into three main categories: fixed-lag, full-smoothing and incremental-smoothing algorithms.

- **Fixed-lag smoothing** estimates all states within a given time window and marginalizes old states to reduce computational complexity. However, marginalizing states outside the estimation window leads to some issues like sparsity, inconsistency, and linearization errors.

- **Full smoothing (batch-optimization)** estimates the entire history of states by solving linear algebraic equations. This framework has the highest accuracy but become infeasible for real-time applications due to trajectory growth over time. However, it includes heavy processing and is not therefore well-suited for resource-constrained systems. Computational costs can be reduced, e.g, by using keyframes, sliding window, or incremental smoothing.

- **Incremental smoothing** takes advantage of the benefits of factor graphs to maintain the level of sparsity, while tackling the full-smoothing and fixed-lag issues. if only a small subset of variables are updated, computational complexity is reduced. Thus identifying and updating variables impacted by new measurements, it is possible to optimize computational cost.

3.3 Lidar-based SLAM

A Lidar sensor naturally captures the depth and geometry of the items in the scene, and using Lidar odometry is possible to estimate the orientation and position of the robot. To identify objects in the environment it would use the distances and shapes returned to the lidar sensor: it tracks laser speckle patterns reflected from surrounding objects. In fact, a lidar-based SLAM combines the visual data and creates a map using edges and planes recorded by the device as features rather than computing adjacent pixels. LiDARs are insensitive to ambient lighting and low-texture environments. The LiDAR-based sensing process involves two main parts: laser emission and optical observation. Laser emission emits coherent and spatial light to the environment, while optical observation reflects laser speckles on the 2D observation plane. Laser reflections are monitored using optical detectors, and a 3D image is reconstructed by contrasting different 2D images.

Lasers are employed in applications with high-speed moving vehicles like self-driving cars and drones because they are substantially more precise than cameras, ToF, and other sensors. Laser sensors often provide point clouds in 2D (x, y) or 3D (x, y, z) dimensions. For the purpose of creating maps using SLAM, the laser sensor point cloud offers highly accurate distance measurements. This method, as compared to cameras, is better for producing accurate 3D digital twin copy of the map. In general, movement is calculated by matching the point clouds in a sequential manner. The vehicle is localized using the determined movement (distance traveled). Iterative closest point (ICP) and normal distributions transform (NDT) methods are two examples of registration algorithms that are used for matching lidar point clouds. Grid maps and voxel maps can be used to visualize 2D or 3D point cloud maps.

The iterative closest point (ICP) method is commonly employed when LiDAR scanning rate exceeds extrinsic motion to calculate a moving object's velocity, addressing motion distortion from single-axis 3D LiDAR. ICP is a standard 3D reconstruction algorithm that iteratively computes correspondences between cloud points of two scans, minimizing the distance between corresponding points. The point-to-plane variant of ICP is a method for reconstructing 3D surfaces by minimizing the squared distance between a point and its tangent plane. This improves performance and accuracy. The GICP (generalized ICP) assumes all measured points are drawn from the Gaussian center at the true point, and maximum likelihood estimation is used for alignment. LiDAR scanning can be slow, so other sensors like cameras and IMUs are often used for velocity measurements. Another approach is 2-axis LiDAR scanning without additional sensors, using laser intensity returns to create visual images and recover motion.

LiDAR odometry is a challenging method due to its resource-constrained nature and the difficulty in accurately detecting motion distortion from objects like glass.

Iterative optical matching and complex computations are required for accurate scans, resulting in poor overall performances.

Moreover, point clouds do not always offer enough information for matching because they are not as highly detailed as photos. For instance, aligning the point clouds in areas with few obstacles might be challenging, which increases the risk of losing track of the vehicle's location.

There are still issues to be resolved, as localization for autonomous vehicles may need to combine data from the global navigation satellite system (GNSS), IMU, and wheel odometry. While 2D lidar SLAM is frequently used for applications like warehouse robotics, 3-D lidar point cloud SLAM can be used for UAVs and automated driving.

3.4 EKF-SLAM

3.4.1 Inroduction to EKF

An EKF is a nonlinear version of the general Kalman filter, which performs first-order linearization around the transition function at each time step. An approximate nonlinear filtering, such as an EKF or particle filter (PF), is used for fusion in nonlinear systems. EKFs provide accurate estimates for Gaussian models with limited linearity, while PFs are suitable for non-Gaussian and nonlinear systems. EKFs are used in robotics due to their computational efficiency.

The core of the SLAM procedure EKF-based VIO computes a robot's position and orientation by determining state propagation from noisy IMU measurements, odometry data, landmarks observations, and correlation from key information extracted from images captured by a single camera or multiple cameras mounted on the robot.

The EKF framework consists of three main steps: state representation, building a measurement model, and an update step.

1. **State representation:** the EKF keeps track of an estimate of the degree of uncertainty in both the position of the robot and the degree of uncertainty in the landmarks it has observed in the environment.

2. **Build a measurement model:** given the robot's new position, landmarks are extracted again. The robot therefore tries to associate observations of these landmarks with observations of previous landmarks it has already seen.

3. **Update position:** The robot's position in the EKF is then updated using re-observed landmarks. New landmarks that have not yet been observed are added to the EKF so they can be re-observed later.

Although EKF framework is one of the most popular filtering strategies, it has some disadvantages: it is difficult to implement in practice and it is not a very reliable method for highly nonlinear systems.

3.4.2 EKF model

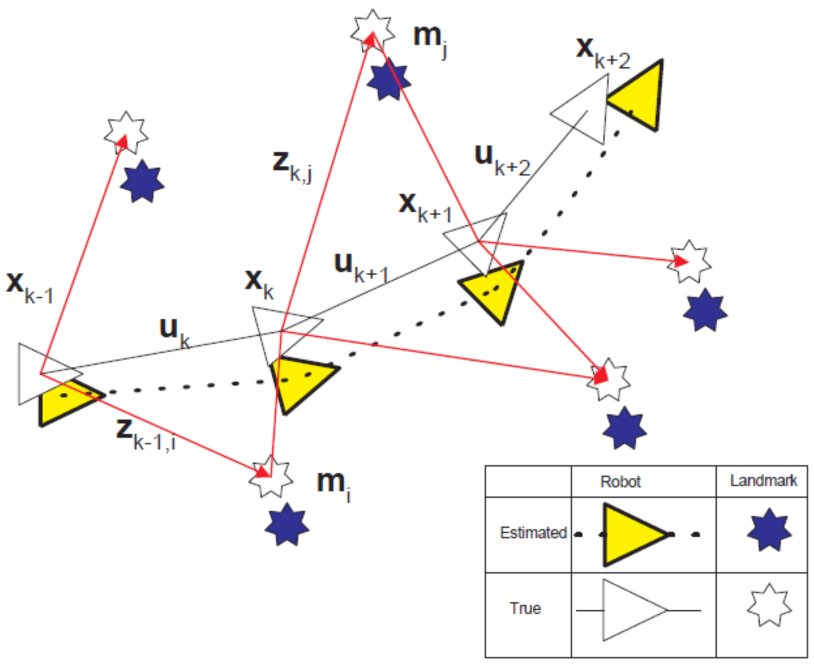


Figure 3.14: The essential SLAM problem. A simultaneous estimate of both robot and landmark locations is required. The true locations are never known or measured directly. Observations are made between true robot and landmark locations.

As explained [19] consider a mobile robot moving through an environment taking relative observations of a number of unknown landmarks using a sensor located on the robot as shown in Figure 3.14. At a time instant k, the following quantities are defined:

- x_k: The state vector describing the location and orientation of the vehicle.
- u_k: The control vector, applied at time $k-1$ to drive the vehicle to a state x_k at time k.
- m_i: A vector describing the location of the i^{th} landmark whose true location is assumed time-invariant.

- z_{ik}: An observation taken from the vehicle of the location of the i^{th} landmark at time k. When there are multiple landmark observations at any one time or when the specific landmark is not relevant to the discussion, the observation will be written simply as z_k.

In addition, the following sets are also defined:

- $X_{0:k} = \{x_0, x_1, \dots, x_k\} = \{X_{0:k-1}, x_k\}$: The history of vehicle locations.
- $U_{0:k} = \{u_1, u_2, \dots, u_k\} = \{U_{0:k-1}, u_k\}$: The history of control inputs.
- $m = \{m_1, m_2, \dots, m_n\}$: The set of all landmarks.
- $Z_{0:k} = \{z_1, z_2, \dots, z_k\} = \{Z_{0:k-1}, z_k\}$: The set of all landmark observations.

The basis for the EKF-SLAM method is to describe the vehicle motion in the form

$$P(x_k|x_{k-1}, u_k) \iff x_k = f(x_{k-1}, u_k) + w_k \tag{3.1}$$

where $f(\cdot)$ models vehicle kinematics and where w_k are additive, zero mean uncorrelated Gaussian motion disturbances with covariance Q_k. The observation model is described in the form

$$P(z_k|x_k, m) \iff z(k) = h(x_k, m) + v_k \tag{3.2}$$

where $h(\cdot)$ describes the geometry of the observation and where v_k are additive, zero mean uncorrelated Gaussian observation errors with covariance R_k.

The SLAM procedure consists of three steps once landmark extraction and data association are complete:

1. **Utilizing the odometry data, update the current state estimation:** The initial step is extremely simple. Just the robot's controls have been included in the previous state estimation. For instance, the robot might be at point (x, y) with rotation theta, controls (dx, dy), and rotational change θ. The new state of the robot $(x + dx, y + dy)$ with rotation $\theta + d\theta$ is the result of the first step.

2. **Updating the predicted state based on new landmark observations:** The second phase takes the previously observed landmarks are considered. You can determine where the landmark should be by estimating its current location. The innovation is basically the discrepancy between the robot's expected position and its actual position as determined by what the robot can see. The second stage additionally updates each landmark's uncertainty to take into consideration recent changes. The landmark certainty, or the variance of the landmark with respect to the current position of the robot, will increase if the landmark is re-observed from this position with low uncertainty.

3. **Add new landmarks to the current state:** The robot map of the environment is updated with new landmarks in the third step. This is accomplished by taking data on the current location and complementing it with data on the relation between the new landmark and the old landmarks.

Computational Effort Every time an observation is made, the joint covariance matrix and all landmarks must be updated as part of the observation stage. This erroneously implies that computation increases with the number of landmarks in a quadratic way. The EKF-SLAM method has undergone extensive research and development, and real-time implementations with thousands of landmarks have been demonstrated.

Data Association The EKF-SLAM solution's typical formulation is particularly vulnerable to improper landmark association. The 'loop-closure' problem, which arises when a robot returns to re-observe landmarks after an extensive traverse, is particularly challenging. In areas where landmarks are not just simple points and really seem different from various viewpoints, the association problem is aggravated. In actuality, the following issues with data association may occur:

- You might not re-observe landmarks every time step
- An object may be a landmark when you first see it, but you may never again see it
- You can mistakenly link a landmark to one you've already seen

Non-linearity Since EKF-SLAM uses linearized nonlinear motion and observation models, it inherits some constraints. In EKF-SLAM, non-linearity can be a serious issue that inevitably results in solutions dramatically inconsistent. Convergence and consistency can only be guaranteed in the linear case.

3.4.3 Other Kalman Filter type

Multi-state constraint Kalman Filter

EKFs are problematic due to their high computational power, making them unsuitable for resource-constrained systems like drones. They have a computational complexity of O(N3) for an N number of features. Structureless methods like MSCKF offer better precision and consistency due to their less strict probabilistic assumption and delayed linearization. MSCKFs also have a linear complexity due to marginalizing 3D feature positions from the state vector. However, the conventional MSCKF algorithm has incorrect observability properties, leading to inconsistency in performance and large estimation errors. [17]

Unscented Kalman Filter

The UKF is a nonlinear Bayesian filter that updates system states using a set of weighted sigma points derived from the prior distribution. UKFs perform statistical local linearization, leading to higher accuracy and third-order linearization, making them superior to EKF-based frameworks. They avoid computing the Jacobian matrix, making them derivative-free. However, the main drawback of the UKF framework is its increased computational power, making its implementation in resource-constrained systems difficult. The main source of nonlinearity is the kinematics of rotation, which is typically modeled using Euler angles or Quaternions.

3.5 UWB

An ultrawide-band transmitter is any transmitter that has a fractional (relative) bandwidth B_r of larger than 20% or a UWB bandwidth B greater than $500MHz$, regardless of the fractional bandwidth, according to the IEEE 802.15.4a standard and FCC guidelines.

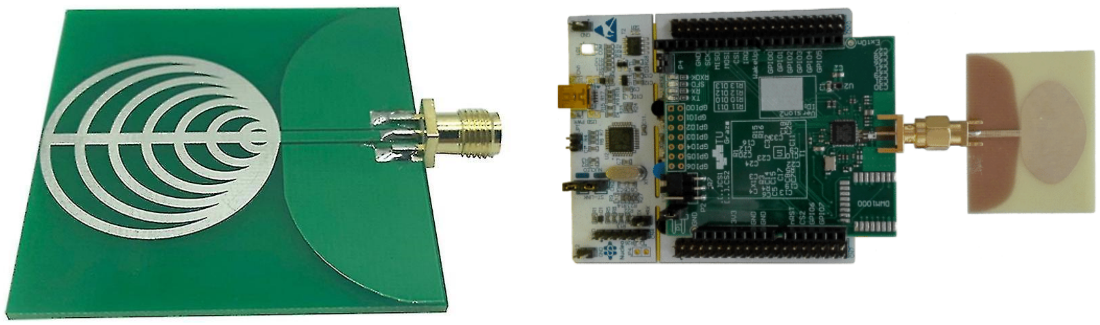


Figure 3.15: Example of a UWB antennas

The frequency range in which the signal has a power spectral density of $10dB$ below its maximum is known as the UWB bandwidth. It is defined as the range between the lower frequency f_L and the upper-frequency f_H.

$$\begin{gathered} B > 500Mhz \\ or \\ B_r = 2\frac{f_H - f_L}{f_H + f_L} > 20\% \end{gathered} \tag{3.3}$$

As we can observe in figure 3.16 cit The UWB technology uses a larger frequency range (from 3.1 to 10.6GHz) at a lower power density than other technologies, which uses narrow bandwidths and high power densities.

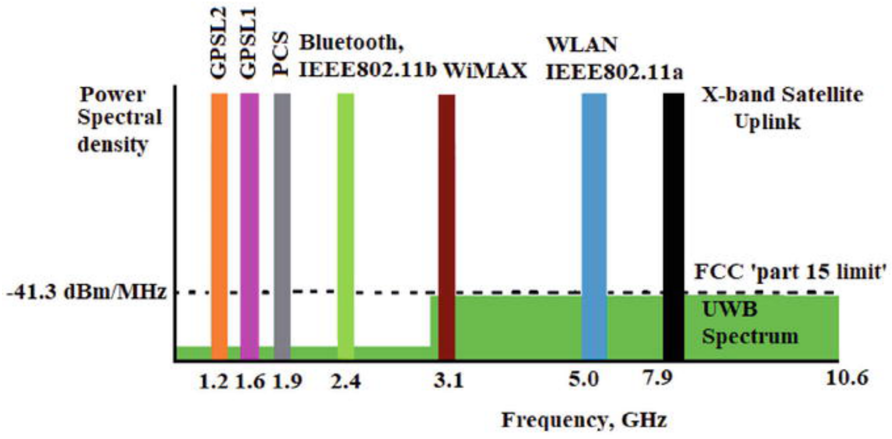


Figure 3.16: UWB frequency spectrum

3.5.1 UWB advantages

A number of benefits that UWB technology provides opens up a wide range of options for different applications. The large spectrum that can be utilized provides considerable design flexibility, enabling the system to be customized to specific requirements also because of the small antenna dimensions. Wide bandwidth allows obstacle penetration, robustness against multi-path propagation, and high data rate transmission (up to 1Gbps) over small distances (less than 1 m). The potential for a range of applications is the second important implication of good time resolution. The extreme shot duration of pulses enables effortlessly obtaining high accuracy (under 10 cm).

The quality of service, data rate, range, and other variables can be adjusted and configured for the specific application. However, a reduced data rate could be compensated for an extended transmission distance: similar to the way data rate and range can be tuned to have an appropriate tradeoff between power consumption, low spectral power increases and interference resistance. The good temporal resolution of UWB is another advantage that makes it robust to multipath propagation and an excellent feature for ranging applications (especially for indoor applications with a lot of interferences and obstacles). and on the tuning of the ranging antennas used. The cost of a single device ranges from around a few dozen to approximately 200 euros, so depending on the application and the scenario it is possible to choose the proper hardware.

3.5.2 How does UWB localization work?

The position of each tag must be determined using anchors, which are emitters with known positions. Therefore, the idea is to properly position these antennas around the warehouse in order to maximize the accuracy minimizing interferences from raks and the building's structure as well as the number of used antennas. Tags are the devices to be tracked, and are installed on the target: in our case, they should be mounted on the drone (or the support AMR). There are two important aspects to keep in mind during the design of a UWB localization system: the number of anchors and their placement.

It is necessary that $m \geq d+1$ where d is the number of physical tag dimensions to be determined and m is the number of available antennas. Therefore in order to get the position of the tag in 2D, are needed at least three anchors while to determine the position in 3D the minimum number of anchors required is four. Positioning accuracy grows along with the system's redundancy as the number of anchors grows. It is often advised to have more anchors than the minimum in order to reduce NLOS (non-line-of-sight) effects.

Anchors should never be positioned on the same line for 2D positioning, or on the same plane for 3D positioning, in order to obtain accurate tracking. Firstly, let's define the term Geometric Dilution of Precision (GDOP). It is defined as:

$$\frac{\Delta(OutputLocation)}{\Delta(MeasuredData} \tag{3.4}$$

It describes the impact of a calculation error on the target position error: the position's reliability decreases as GDOP increases. When anchors are positioned on the same line in 2D positioning or on the same plane in 3D positioning, it tends to infinity. When the tag to track is outside the area delimited by the anchors (such as during the chase phase), the GDOP grows with increasing distance from it.

There are different techniques to estimate the distance of two UWB antennas and then the system position. Let's see the main ones.

Time of arrival (TOA) The potential of UWB technology is generally best exploited through time-based ranging approaches, as can be illustrated mathematically through the Cramer-Rao inequality.

$$\sqrt{Var(d) \geq \frac{c}{2\sqrt{2}\pi\sqrt{SNR}\beta}} \tag{3.5}$$

where the first member is the accuracy of the estimation of the distance d, c is the speed of light, SNR is the signal-to-noise ratio and β is the effective signal bandwidth.

The so-called Cramer-Rao lower bound for a single path additive white Gaussian noise channel for time-based ranging is formulated using this equation. Given the properties of the signal, this is the maximum localization accuracy that can be obtained. It's important to note that the bandwidth at the second member of the inequality's denominator, confirms that time-based UWB ranging systems can reach very high accuracy in distance measurements. Considering a line-of-sight (LOS) condition, where the first path is the direct path of the signal, measuring the first path delay τ_0 it is possible to compute the distance d between the transmitter and the receiver as $d = c \cdot \tau_0$ where c is the speed of light. For a narrow band system, given the transmitted signal $s(t)$, the received signal is expressed as

$$r(t) = As(t - T) + n(t) \tag{3.6}$$

where the parameters A and T are affected by all the multipath components and $n(t)$ is the white noise. The matched filtering estimator can only measure the optimal value of T which includes the effects of the reflected signals so it is different from the first path delay.

Once the signal's arrival time is measured at the receiver, the Time of Flight (TOF) can be computed as the received message contains the starting time. The sender and receiver's clocks have to be perfectly synchronized for this method to work. The Cramer-Rao lower bound, expressed by Equation 3.5, shows that the achievable accuracy under ideal conditions is really high, thanks to the very high bandwidth of the pulse signals. This means that precise clock synchronization between nodes becomes a crucial issue in obtaining accurate distance estimation using TOA. In order to get around the problem of clock synchronization, TOF measurement techniques like Single-Sided Two Way Ranging (SS-TWR) and Double-Sided TWR (DS-TWR) have been developed see Figure 3.17. In fact, these protocols are compared only by time measurement taken on the same device, avoiding the necessity of clock synchronization. In SS-TWR the TOF is:

$$T_{tof} = \frac{1}{2}(t_{roundA} - t_{replyB}) \tag{3.7}$$

where $t_{roundA} = \tau_{AR_x} - \tau_{ATx}$ is the actual round-trip time of a signal measured at Device A and $t_{replyB} = \tau_{BTx} - \tau_{BRx}$ is the actual reply time of a signal measured at Device B. In the DS-TWR, the measurements are computed on both sides, requiring a further reply message from the device that started the TWR, so we have:

$$T_{tof} = \frac{1}{4}[(t_{roundA} - t_{replyB}) + (t_{roundB} - t_{replyA})] \tag{3.8}$$

TWR is still subject to problems including propagation-time delay, transmission-time delay, receiving-time delay, and preamble accumulation-time delay even if the clock synchronization error is prevented.

Time Difference of Arrival (TDoA) is a variant of ToA that can be used to determine a transmitter's location when synchronization between the mobile unit and the anchors cannot be ensured. In this scenario, ranges are not directly

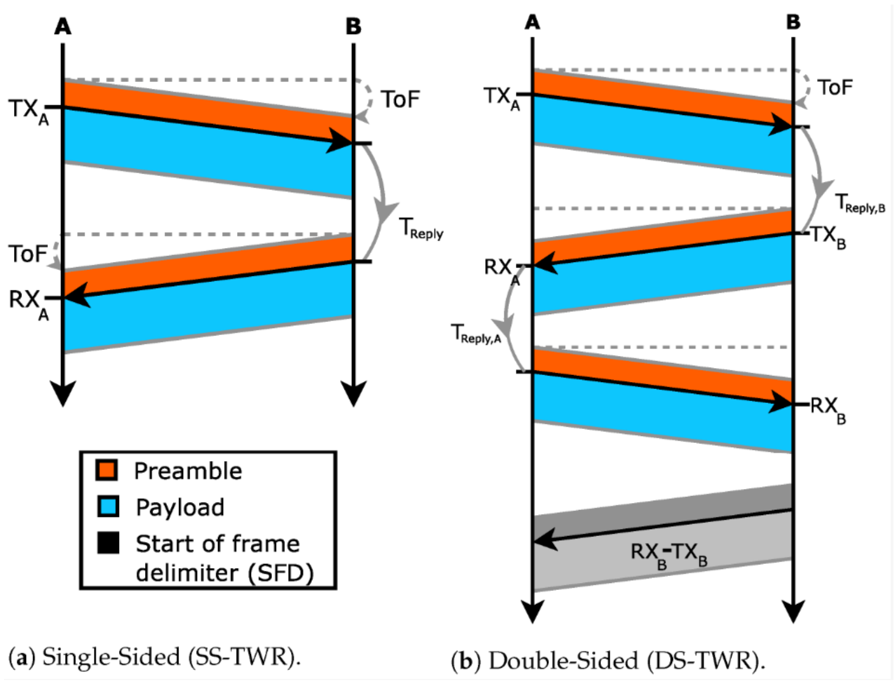


Figure 3.17: Principles of single-sided two-way ranging (SS-TWR) and double-sided two-way ranging (DS-TWR) protocols

measured; instead, the ToA of an emitted pulse is measured at several known-position receivers.

Angle of Arrival (AoA) is a localization technique where it is measured the angle of arrival of a signal instead of its distance. This type of measurement is more complex and it requires more complicated hardware, like an antenna array. The advantage of this method is that only two measurements are required to identify a position in a 2D.

Received Signal Strength (RSS) As can be imagined, a signal weakens as it propagates further from its source. Knowing the transmitted power, this phenomenon can be used to obtain an approximation of the distance between the transmitter and the receiver. Despite being simple to implement, the accuracy of this method is very low, especially in enclosed spaces where free space propagation cannot be assumed. Making a map of the surrounding environment, known as a "fingerprint," is a way to solve this issue. The position of the mobile device is then determined by comparing the strength of the received signal from numerous antennas to the map that was made earlier. It is important to note that this procedure takes a lot of time and must be repeated for each new configuration of the localization system.

Multilateration and pose estimation

It is feasible to determine the target's position by computing the distances between the target node and the anchor nodes whose location is known. For the sake of simplicity, let's start with the 2D case with three anchors, as the method is easily applicable to higher dimensions and larger UWB range devices see Figure 3.18. The intersection of the circles whose centers are the locations of the anchors and the radius are the distances between them and the tag provides the exact position in the ideal scenario when the distances are not impacted by errors. The system is overdetermined and the solution remains unchanged if more anchors are provided. In the 3D case it is sufficient to intersect at least four spheres following the same procedure (Figure 3.19. The unknown coordinates are calculated mathematically

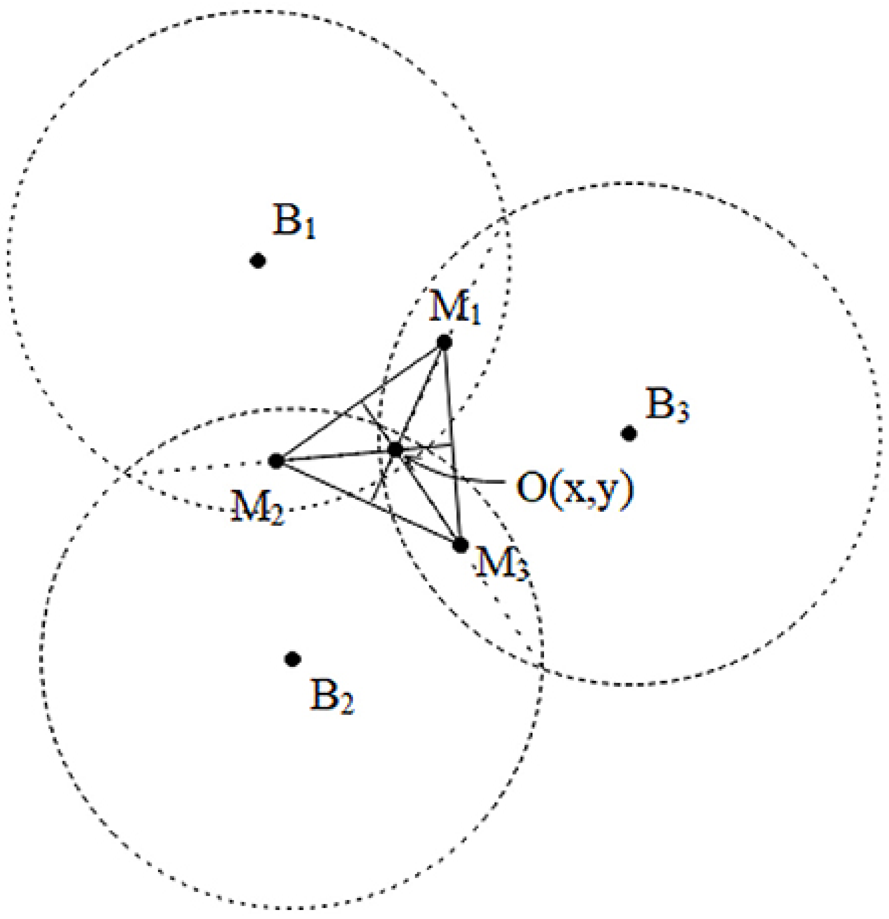


Figure 3.18: UWB 2D positioning: geometrical interpretation of multilateration

as a solution to the circle-or-sphere system of equations, which is overdetermined whenever the number of anchor nodes is greater than the minimum. Furthermore, due to the presence of intrinsic mistakes in ranging measurements, it is quite normal scenario that the circles do not all intersect at the same point. Therefore, using numerical algorithms like Linear Least Squares (LLS) and the Gauss-Newton (G-N) that minimize the error function between the real and estimated position is possible to estimate the position and reduce the error. The first is easier to develop and more appropriate in the 2D scenario, whereas the second was chosen because it offers a very good trade-off between computation time and accuracy and is simple to use. Therefore, in addition to range issues, the spatial deployment

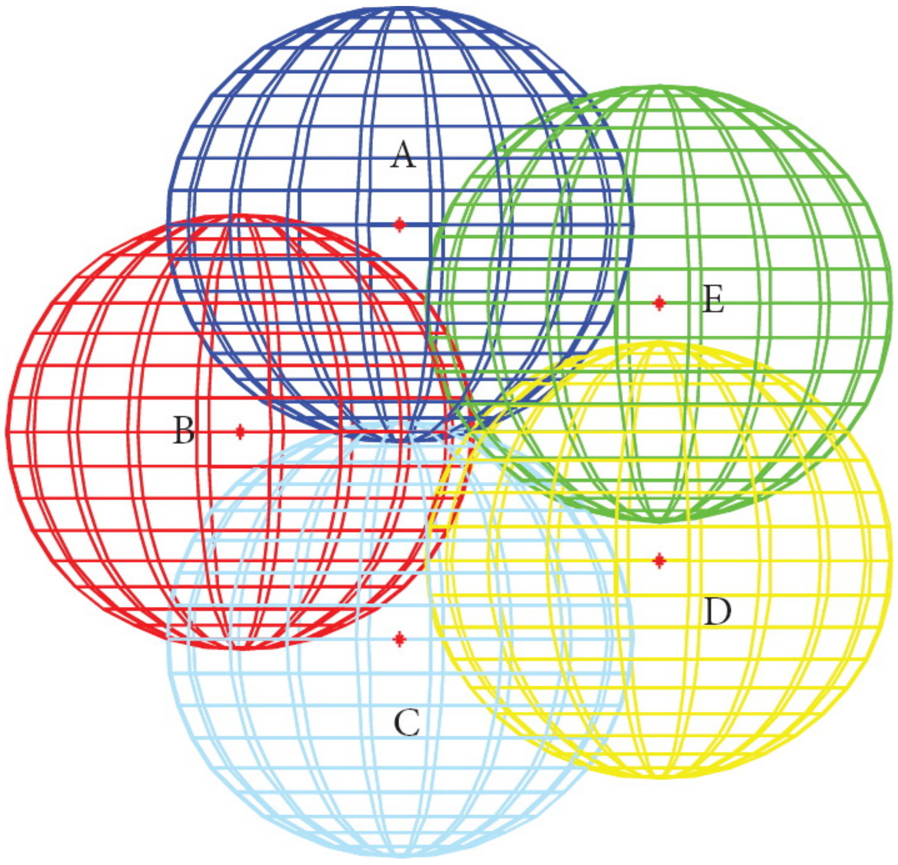


Figure 3.19: UWB 3D positioning: spheres intersection

of the anchors is another important element that affects the localization accuracy. While certain positioning algorithms are more sensitive than others, poor sensor placement can always result in singularities and inaccurate position estimation. For example, heuristic search approaches, acute triangular-based deployment, adaptive beacon placement, and optimal placement solutions using the maxL-minE algorithm have all been suggested as ways to reduce localization errors in the working area. When there are several anchor nodes accessible, it has been found that selecting a subset of the sensors appropriately, produces better results than using them altogether. Convex hull selection, entropy-based information gain, and joint clustering methodology are some of the methods to carry out such selection. Nevertheless, these methods work well when the anchors are considered to be fixed, which is the case for almost all UWB localization systems.

3.5.3 Main sources of error

The theoretical ranging accuracy achievable with time-based ranging algorithms is quite high, but only in a single user, LOS, and single path scenario. In real-world scenarios, there are a number of difficulties that affect the localization system's accuracy: here are presented the most significant UWB positioning causes of error.

Multipath propagation: The time shift of the template signal that creates the highest correlation with the received signal is often used to estimate TOA. Traditional correlation-based algorithms can be employed when dealing with single-path channels, and the transmitted signal is used as the best template signal. However, in actual use, the channel is impacted by multipath propagation, and the template signal is produced by convolutioning the transmitted signal with the impulse response of the channel. Thus, the correlation between the received signal and the transmit-waveform template is sub-optimal and, if applied to narrow-band signals, it often gives wrong TOA estimates because of the overlapping of the transmitted signal and the reflected ones. [20]

Multiple access interference: Signals from several nodes can interfere when multiple users are operating in the same space. This results in poorer TOA estimation and, thus, less accurate positioning. Different methods, such as pulse-based polarity randomization and time-hopping codes with low cross-correlation features, can be used to solve this problem. [20]

NLOS propagation: Non-line of sight conditions occur when an obstacle between the transmitter and receiver prevents the reflected signal from reaching the receiving node, causing a positive bias in distance measurement. To achieve optimal localization accuracy, the typical approach excludes NLOS measurements, requiring differentiation between LOS and NLOS components. Techniques in literature often use range statistics (LOS estimated range is Gaussian distributed with zero mean whereas NLOS are biased and non-Gaussian) or channel characteristics, such as received signal power and features extracted from the power delay profile. In some cases, LOS measurements are insufficient, and environmental information can compensate for NLOS errors. A combination of LOS and environmental information can be used in such cases.

High time resolution: UWB signals have a large bandwidth, allowing for high time resolution and accurate TOA estimation. However, practical systems face challenges such as clock jitter, which affects TOA estimates due to clock accuracy and drifts in target and reference nodes. Additionally, sampling the received signal above the Nyquist rate is impractical, making TOA estimation schemes use frame-rate or symbol-rate samples for low-power designs.

3.6 Wheel odometry

Wheel odometry is a technique used to estimate motion and position in wheeled robots and autonomous vehicles. It is done by using rotary encoders, sensors attached to the motors of the wheels, to measure rotation. Differential drive robots have separate motors controlling each wheel, which can spin the wheels at different speeds and directions, allowing in this way, the robot to move easily in all directions. The data for the odometry model will come from rotary encoders, which attach to the motors and collect rotation data. Two rotary encoders, one attached to the left wheel's motor and another to the right wheel's motor, will be used to measure the distance traveled by the wheel.

The rotation data, along with information on the encoder, such as the radius or circumference, can be used to compute and estimate the distance traveled by the wheel. Knowing the number of slits passed can help determine the amount of rotation between time steps. For an optical encoder, where all slits are equally spaced, the total angle of rotation between time steps can be obtained by multiplying the number of slits passed by the amount of rotation represented by a single slit. The angle of rotation can then be multiplied by the encoder's circumference to get the distance traveled by the wheel.

3.6.1 Incremental encoders vs absolute encoders

Absolute encoder An absolute encoder is a device that maintains position information even when power is removed from it, ensuring the accuracy of the system. The encoder's position is available immediately upon applying power, and the relationship between the encoder value and the physical position of the controlled machinery is established at assembly: the system does not need to return to a calibration point to maintain position accuracy. The optical encoder's disc is made of glass or plastic with transparent and opaque areas, it is read by a light source and photodetector array, often using the Gray code. The absolute analog type produces a unique dual analog code that can be translated into the absolute shaft angle. A parallel absolute encoder has multiple code rings with binary weightings, while a multi-turn absolute rotary encoder includes additional code wheels and toothed wheels. High-resolution wheels measure fractional rotation, while lower-resolution geared code wheels record the number of whole revolutions of the shaft.

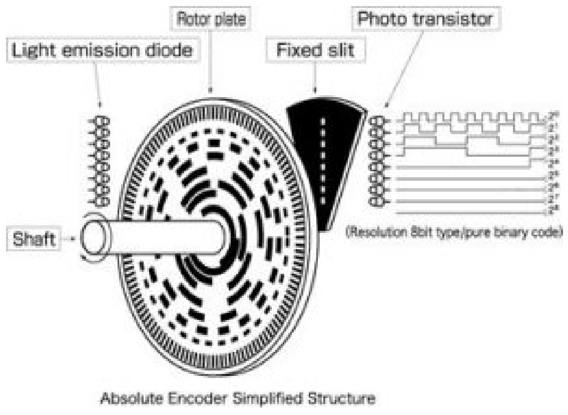

Rotor plate
Photo transistor
Light emission diode
Fixed slit
Shaft
(Resolution 8bit type/pure binary code)
Absolute Encoder Simplified Structure

Figure 3.20: Absolute rotary encoder

Incremental encoder An incremental encoder is a device that reports changes in position, which is crucial in some applications but does not track absolute position. This means that the mechanical system monitored by an incremental encoder may need to be homed to initialize absolute position measurements. The rotary incremental encoder is the most widely used of all rotary encoders due to its ability to provide real-time position information. Its measurement resolution is not limited by its internal movement sensors, and it can have up to 10,000 counts per revolution. Its output signals, A and B, issue a periodic digital waveform in quadrature when the encoder shaft rotates, combining the characteristics of an encoder and a resolver. The waveform frequency indicates the speed of shaft rotation, the number of pulses indicates the distance moved, and the A-B phase relationship indicates the direction of rotation. Incremental encoders report position changes without prompting and convey this information at faster data rates than most absolute shaft encoders.

A rotary incremental encoder uses mechanical, optical, or magnetic sensors to detect rotational position changes. Encoders with mechanical sensors require switch debouncing and consequently are limited in the rotational speeds they can handle, for this reason are commonly used as digital potentiometer control on electronic equipment, while optical encoders are used for higher speeds or precision.

Incremental optical encoders use a light-emitting diode (LED), a disk with slits, and a circuit with a photo sensor. The disk separates the LED and circuit with a photo sensor and as the motor spins, the disk rotates, allowing light from the LED to pass through the slits to the photosensor, changing the circuit's voltage. The number of times the voltage changes corresponds to the number of slits passed, providing information on the angle of rotation. This incremental encoder allows for the measurement of rotational position changes, unlike absolute encoders, which determine the motor's exact orientation at each measurement.

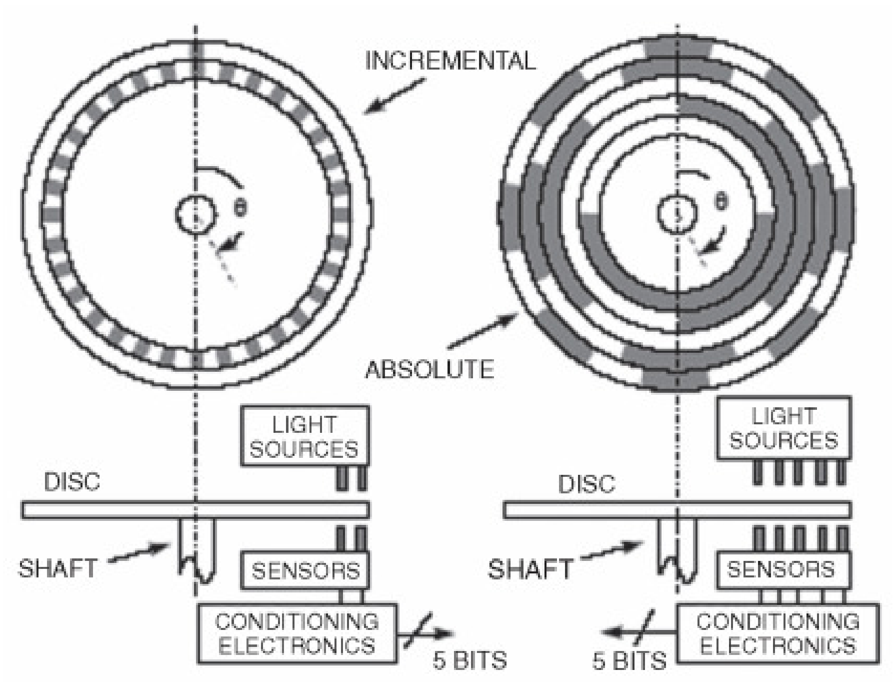


Figure 3.21: Comparison between incremental and absolute encoder

3.6.2 A simple model of wheel odometry

This model aims to estimate the position and orientation of a robot using data from rotary encoders, robot dimensions, and geometry[21]. The encoder provides information on wheel distances at each time step. The robot's dimensions are represented as a point, and the distance from the left and right wheels is the only dimension needed. The reference point is located equidistant between the two wheels, requiring only one number to be tracked.

Let's define some variables to keep track of these ideas:

$$d_{L,t} = \text{distance traveled by the left wheel at time t (encoder)}$$

$$d_{R,t} = \text{distance traveled by the right wheel at time t (encoder)}$$

$$d_w = \text{distance between the reference point and the wheels (dimension)}$$

$$d_L = \text{distance traveled by the left wheel}$$

$$d_R = \text{distance traveled by the right wheel}$$

$$d = \text{distance traveled by the reference point}$$

$$\Delta\theta = \text{change in the angle of rotation}$$

$$R = \text{radius of the curve containing the reference point}$$

The first two variables correspond to the distance traveled by the wheel at a certain time step. This information will come from our rotary encoder. The third variable d_w can be derived by measuring the distance between the two wheels and dividing it in half since the point is equidistant from the two wheels, or measured with a ruler. The last three variables are not directly measurable but we need to use geometry to relate these variables to the measurable quantities.

The robot's motion is modeled as a curve along a circle, which allows for the calculation of distance traveled and orientation angle. This model encapsulates straight motion, that corresponds to a curve with a small angle and/or large radius, with the curvature decreasing as the angle or radius increases.

To encapsulate rightward motion in the model the idea is to flip the diagram horizontally, revealing symmetry. The variables associated with the left wheel and right wheel would flip in sign, resulting in a positive to negative, negative to positive orientation estimate. However, the distance estimation remains the same.

Backward motion, which involves distances going in the negative direction, is captured through negative distance values. The key variables of interest are the distance traveled by the reference point and the change in the angle of rotation. The radius of the curve containing the reference point is not needed anymore, as it is no longer useful for derivation.

In summary, the robot's motion is modeled as a curve along a circle, with the key variables being the distance traveled by the reference point and the change in the angle of rotation. Thus, the key results from our model so far are:

$$d = \frac{d_L + d_R}{2}$$

$$\Delta\theta = \frac{d_R - d_L}{2d_w}$$

The results indicate that the distance traveled by the reference point and the change in orientation between time steps are relative, but the direction and new orientation of the robot are not known.

To simplify the model, the distance traveled by the reference point is represented as a line instead of a curve. This is because, in wheel odometry with encoders, the data sampling is high, resulting in a small time window between measurements and minimal motion captured by each time step. This results in a small curvature of the arc, resembling a straight line. The distance is now represented as a straight line, thus is a safe assumption and simplification. The addition of the previous time step's orientation doesn't change the distance traveled by the reference point or the change in angle of rotation, as the formulas derived from earlier don't rely on the orientation angle. Instead, the robot's orientation changes from relative to absolute on the coordinate plane. Thus, the absolute orientation angle at any time step can be defined by:

$$\theta_t = \theta_{t-1} + \Delta\theta_t$$

When working with absolute motion, our robot will have a coordinate point at each time step. The robot's coordinate position is updated using trigonometric properties, where cosine and sine of an angle are adjacent and opposite. The distance traveled by the reference point, the angle of orientation from the previous time step, and the angle resulting from motion can be calculated. Adding the x and y distance to the previous time step's coordinate, we can determine the new coordinate position of the robot. We can describe the dynamics with these equations:

$$x_t = x_{t-1} + dcos(\theta_{t-1} + \frac{\Delta\theta_t}{2})$$

$$y_t = y_{t-1} + dsin(\theta_{t-1} + \frac{\Delta\theta_t}{2})$$

For absolute motion, we have the coordinate position, comprised of an x and y component, and the absolute orientation angle (in radians). The three equations to define the odometry model for absolute motion are expressed in vector form:

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{bmatrix} + \begin{bmatrix} d_t cos(\theta_{t-1} + \frac{\Delta\theta_t}{2}) \\ d_t sin(\theta_{t-1} + \frac{\Delta\theta_t}{2}) \\ \Delta\theta_t \end{bmatrix}$$

3.6.3 Wheel odometry issue

The wheel odometry approach has some limitations. For instance, it cannot be used with aerial or aquatic vehicles but only with ground vehicles. Additionally, it experiences a position drift phenomenon, in which measurement errors increase over time. Wheel slippage also causes wheel odometry systems to perform poorly on complicated uneven terrains and slippery surfaces. Wheel odometry is a straightforward and affordable localization method, but it is inadequate for controlling platforms that need a precise and durable localization system. [17] For these reasons, wheel odometry is often associated with fiducial markers and anchors that reset the drift error from time to time.

3.7 Fiducial Marker relocation

As reported in [22] a fiducial marker is an important tool when using a single camera for position estimation in industrial systems, augmented reality, robot navigation, or human-robot interaction. These markers are used for identifying a specific object. Normally, this can be difficult because some objects have incredibly complex shapes or, under certain conditions, don't have enough contrast to be visible for single-camera pose estimation. When an object has a fiducial marker attached to it, it is much simpler to identify the object. In general, these objects are used to relocate the robot's pose, in fact, once a marker is identified by a camera, the camera's relative position with respect to the marker can be estimated based on how the camera perceives the specific marker thanks to particular PnP (Perspective-n-Point) algorithm that are explored in the following paragraph.

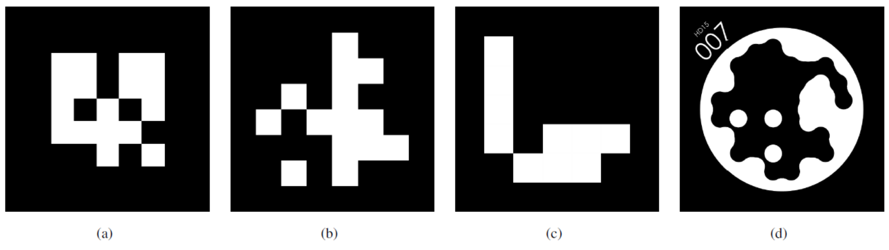


Figure 3.22: Example of fiducial marker: (a) ARTag, (b) AprilTag, (c) ArUco, and (d) STag

There are many different types of markers; the majority of them are square and estimate pose using the marker's corner points. Additionally, there are round or other shaped markers like INTERSENSE and REACTIVISION that work differently and can be used in other contexts. The most suitable marker for object localization and tracking are the square and planar markers as ARToolKit, ARTag, ARToolKit+, AprilTag, STag CALTag and ArUco, in fact for this project will be used ArUco and AprilTag.

AprilTag Similar to ARToolkit+, AprilTag was released under an Open Source license, resulting in particularly well-documented algorithms and implementations. With AprilTag, tags can be detected quickly and accurately because only 4 bits (black or white squares) of data are encoded by the smallest markers. The system is characterized by an increased number of different codes and an increased number of bit errors that could be detected, despite having 36 bits for the largest markers.

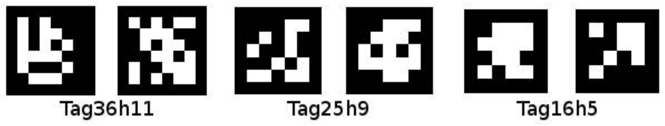


Figure 3.23: Example of different AprilTag

ArUco At first glance, the AprilTag and ARToolkit+ are very similar to the ArUco marker. The main difference between these two is that the intention is to create a dictionary with the greatest inter-marker distance. This inter-marker distance is used to identify and fix false-positive results. However, this AruCo marker's computation time is faster than that of AprilTag and ARToolkit+.

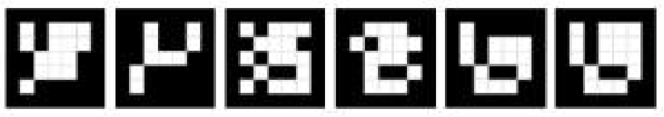


Figure 3.24: Example of different ArUco

How to estimate the pose?

Perspective-n-Point Perspective-n-Point is a problem that estimates the pose of a calibrated camera given n 3D points in the world and their corresponding 2D projections in the image. The camera pose consists of 6 degrees of freedom (DOF) which are made up of the rotation (roll, pitch, and yaw) and 3D translation of the camera with respect to the world. This problem originates from camera calibration and has applications in computer vision, robotics, and augmented reality. A commonly used solution is P3P for n = 3, and many solutions are available for n $\geq$ 3. In our case as reported in [23] the rotation matrix and translation matrix can be used to describe the pose of an object in relation to the camera coordinate system for an image of that object that was captured by the camera. The translating matrix T can be used to determine the separation between the object in the image and the camera lens.

$$s * p_i = A[R_G|T_G]q_i \tag{3.9}$$

Where s is the required scale factor, $_pi$ is a point in the image, and q_i is the 3D coordinate of the point p on the image corresponding to the camera coordinate system.

The relationship between the camera coordinates and the ideal coordinate system without image distortion is expressed by matrix A, which represents the internal parameters of the camera. $[R_G|T_G]$ is the external parameter, which indicates the position and orientation of the camera in the GLOBAL coordinate system; R is the rotation matrix, and T is the translation matrix.

$$Tz \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_G \\ Y_G \\ Z_G \\ 1 \end{bmatrix}$$

"The internal parameter A in formula 3.9 can be obtained through the calculation of the camera calibration process. f_x and f_y in A are the focal lengths at the pixel scale of the camera. As long as the coordinates of four known points on the same plane are obtained, the rotation matrix R and translation matrix T can be obtained, and then the distance between the marker and the camera can be calculated." [23]

Rodrigues formula Once the fiducial marker's coordinates are obtained is possible to estimate the position. As explained in [24] fiducial markers are placed on known locations for localization, allowing for global pose estimation and position relocation. In this way, every marker ID has its own known position; thus, it is possible to estimate the pose globally once the camera identifies it in the environment. For example ArUco library provides two vectors: rotation and translation, that represent relative rotation and translation between the camera and marker. The Rodrigues formula converts the rotation vector to a rotation matrix. Each marker has a transformation matrix, transforming the relative position between camera and marker to the global coordinate system. This system allows for accurate localization and accurate pose estimation.

$$R_{CT} = Rodrigues(rvec)$$

Once get R_{CT} - rotation matrix of the camera relative to the tag, it is applied a transposition to find the tag's rotation relative to the camera.

$$R_{TC} = R_{CT}^T$$

where R_{TC} is the rotation matrix Tag relative to the camera. From this, to find the translation vector of the camera (in ArUco coordinate system) t_{cam}^{ar} we have:

$$t_{cam}^{ar} = -R_{TC} * t_{vec}$$

where t_{vec} is the translation vector acquired from ArUco library. At this point, other transformations are needed to get the final NEU (North East Up) conventional coordinate system. In the following formulas, we will have t for the translation vector and q for quaternion.

$$q_{cam}^{ar} = q_{TC} * q_{X\pi} \tag{3.10}$$

where q^{ar}_{cam} is the rotation quaternion of a camera in the AR coordinate system, q_{TC} is the rotation matrix of Tag relative to the camera and $q_{X\pi}$ the rotation quaternion of X axis + 180°.

$$q^{neu}_{cam} = q^{ar}_{cam} * q_{Y\pi/2} * q_{X-\pi/2} \tag{3.11}$$

where q^{neu}_{cam} is rotation quaternion of camera in NEU coordinate system, $q_{Y\pi/2}$ rotation quaternion of Y axis + 90° and $q_{X-\pi/2}$ is the rotation quaternion of X axis - 90°.

$$t^{neu}_{cam} = t^{ar}_{cam} * R_{Y\pi/2} * R_{X-\pi/2} \tag{3.12}$$

where t^{neu}_{cam} is the translation vector of camera position in NEU coordinate system, $R_{Y\pi/2}$ rotation matrix of Y axis + 90° and $R_{X-\pi/2}$ is the rotation matrix of X axis - 90°.

$$t^{glob}_{cam} = T^{init}_{marker} * t^{neu}_{cam} \tag{3.13}$$

where t^{glob}_{cam} is the translation vector of camera position in GLOBAL coordinate system and T^{init}_{marker} is the initial translation of ArUco marker. And finally, we can obtain q^{glob}_{cam} that is the rotation quaternion of the camera in the GLOBAL coordinate system.

$$q^{glob}_{cam} = q^{init}_{marker} * q^{neu}_{cam} \tag{3.14}$$

where q^{init}_{marker} is the initial rotation of ArUco marker and q^{neu}_{cam} is the rotation quaternion of the camera in NEU coordinate system.

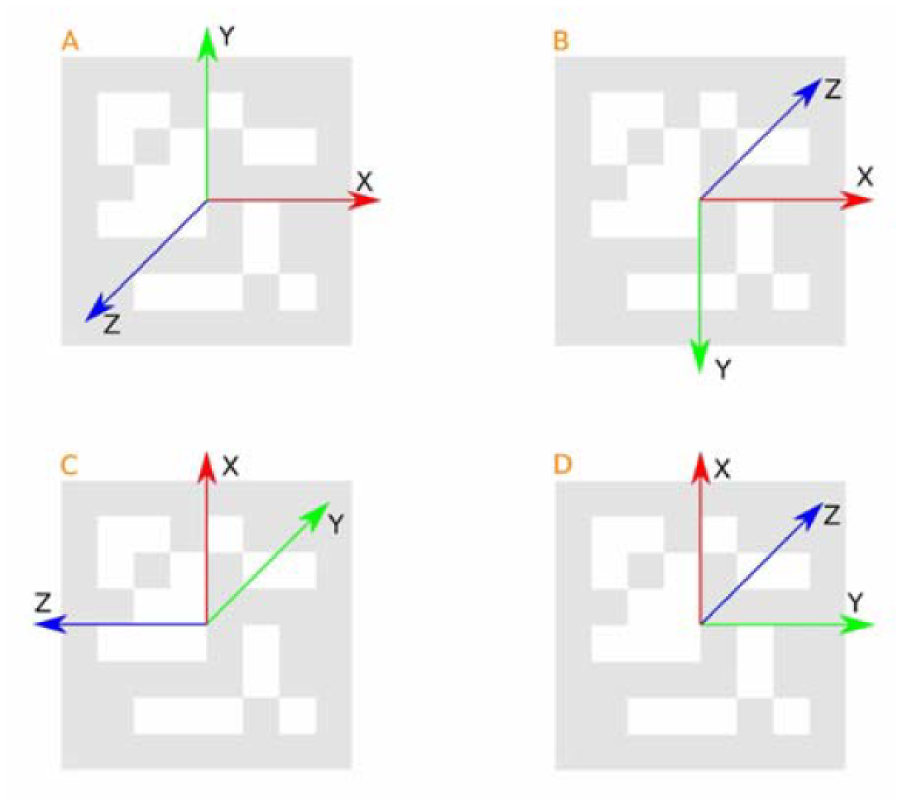


Figure 3.25: Transform of ArUco coordinate system

After equation 3.10 the coordinate system rotates from fig. 3.25 A to fig. 3.25 B. In the same way is transformed also the translation vector 3.12 and finally with 3.13 and 3.14 local coordinate system are transformed in global system coordinate, giving as a result the camera pose estimation relative to a warehouse or indoor environment. Having these coordinates is now possible to relocate the position of the robot based on this new "trust and known" pose, resetting in this way the drift error.

3.8 Conclusion

In this chapter, we have therefore covered the main indoor navigation techniques that are used for autonomous systems today. Most of them can be used on stand-alone drones, while others such as Lidar for example, are not really suitable for use on a stand-alone drone as the sensors may be too heavy, energy-intensive, or may not work properly due to the vibrations of the drone. Finally, other techniques such as wheel odometry by design cannot be used on a standalone drone but only in combination with an AMR.

Some of them are simpler and more production-ready, such as VIO, VO, Lidar-SLAM, and wheel odometry, in fact once these sensors are mounted on the drone or AMR, they are ready to work without needing to prepare the environment or make onerous configurations. This is possible because of the extremely advanced algorithms and technologies available nowadays, just a few years ago it was unimaginable that an object weighing just a few grams could give its position in real-time with very high levels of accuracy and low power consumption.

Naturally, their performance and accuracy are high if the environment conditions they have to work in are adequate: in a dark environment, as we have said, the visual-SLAM can never work properly viceversa if the illumination is too high, reflections and shadows could as well affect the results of the algortims. Similarly also for wheel odometry, is unaffected by lighting conditions, but if the floor has irregularities, steps or unevenness, this technique would also fail. These odometric techniques moreover, as we have seen several times in this chapter, are subject to drift error, which increases more and more with time, so it is necessary to make sure that from time to time the error is reset to zero. It is, therefore, necessary to have an independent, parallel localization system with which to compare the 2 estimated poses, and use sophisticated and complex filters and logic to estimate the final pose.

At this point 2 other techniques that are not subject to drift error, and which are not based on odometric algorithms, come into the equation: the UWB positioning and fiducial marker relocation.

UWB allows very precise continuous real-time localization, with a cm-accuracy, however for indoor environments such as a warehouse it has severe limitations both due to electromagnetic interference but also because it is not a production-ready solution. In fact, it is necessary to install dozens or even hundreds of transmitter antennas inside the warehouse, placing them strategically so that the entire warehouse is well covered. In addition, since this is an active system, it must be powered and maintained, which means that wiring must also be done or alternatively using battery-powered antennas assuming they are sized properly but for sure maintenance will be more frequent and expensive.

Fiducial markers from this point of view appear much more promising since each fiducial marker has basically zero cost, but more importantly, it is a passive system, unlike the antenna, so it does not have to be maintained and installation is much easier and faster. It might be considered that the drawback is the fact that they need proper lighting conditions, it is true, but actually, there are markers made with fluorescent materials that can be seen in the dark, or with materials that can be seen very well in the infrared or others even can be seen very well under direct illumination (light beam spot-on fiducial marker) without generating disturbing and strange reflections. In any case, it must be considered that unless using only Lidar or wheel odometry (which also works at night) as odometric localization, if

VO is to be used, even in this case the light conditions must be good.

Fiducial markers are not a continuous real-time localization system, but a relocation system that is used to reset the drift error to zero. Therefore, in our research, we wanted to focus mainly on VO and fiducial marker relocation in a well-illuminated environment because these 2 techniques compensate each other perfectly. VO is used for navigation within corridors and from time to time the error is reset with fiducial markers strategically placed within the warehouse/lab.

In the following chapter, is presented the work done divided into two phases. In phase 1 it is implemented ROS nodes for the management of VO odometry and relocations based on fiducial markers. Then based on the preliminary obtained results, for phase 2 of this study, it has been decided to focus on a comparison between 2 families of fiducial markers (AprilTag and ArUco) and 4 different types of cameras (stereo/fisheye, wide-angle camera, and two different standard camera with different performances.

Chapter 4

Empirical studies and measurements for sensor fusion optimization

4.1 Introduction and research presentation

The initial idea of the research was to implement a sensor fusion algorithm that by receiving odometry data from different sensors (via ROS nodes), after having processed and filtered them, would produce as output real-time coordinates that would take into account the contribution of the different sources: thus providing a "theoretically" more accurate drone pose (in our case of the testbed) with lower drift error.

Phase 1: IntelRealsense T265 tracking camera was chosen to be used as the source of VIO odometry data and a simple webcam, to perform fiducial marker relocation. The testbed (which simulated the drone) used for testing consists of a wheeled bench (see photo 4.1) and a 3D printer-printed holder on which the 2 cameras were mounted (see photo 4.3). The T265 was mounted so that it pointed forward, with a slight downward tilt (so as not to point exactly to the horizon), so that VIO could also extract even features from the floor and estimate the pose better. The webcam, instead, was installed perpendicular to the floor level in order to identify all fiducial markers specially placed on the floor of the test area (see photo 4.2).

The two cameras were installed at a height of about 158 cm from the floor; this height was chosen considering navigation within warehouse corridors. Excessive height would make it difficult to read markers on the floor, or in any case, inaccurate

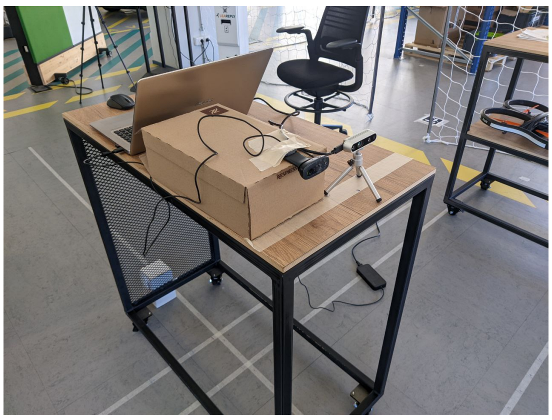


Figure 4.1: A photo of the wheeled bench used to perform all tests

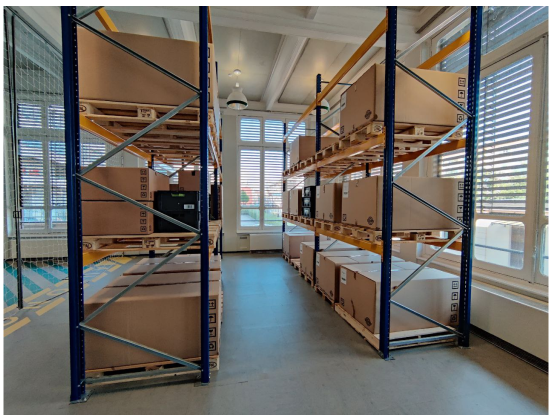


Figure 4.2: Photo of the lab where tests were performed

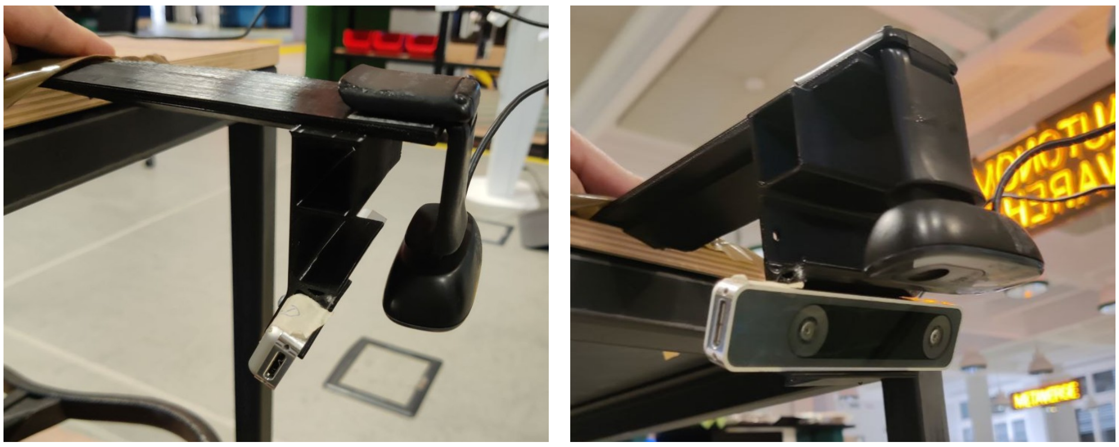


Figure 4.3: Photo of the T265 and Logitech webcam holder

pose estimation, and vice versa if the drone is flying very close to the ground it would have a very limited field of view and the vortices generated by the propellers would interfere with the flight dynamics by generating vibrations that would affect VIO accuracy.

The first test consisted of taking the odometry data from the VIO and comparing them with the pose data obtained from the fiducial markers (when present). From the comparison, it was noticed that the VIO worked very well: the localization was very precise and the drift error very low, while the relocation done with the fiducial markers had a very high error, so it was completely useless if not counterproductive for the purposes of sensor fusion and relocation. Indeed, as we will see later, the generated point cloud was very large surroundings making it impossible to perform an accurate relocation.

Phase 2: At this point it was therefore considered to investigate further the fiducial marker-based relocation theme, in order to try to understand why the accuracy was so low. Hence, it was decided to make a comparison between 2 families of fiducial markers (ArUco and AprilTag) and 4 different types of cameras.

After that, given all the measurements and tests, data were collected to be analyzed and draw some conclusions.

4.2 Software and hardware set up

Various libraries and software (where possible open source) along with sensors, cameras, and mini processors/computers were used in the implementation of this study. The hardware and software used and their setup are presented below. Below is the list of used hardware:

- PC ASUS Vivobook X580GD
- Nvidia Jetson Nano
- Intel Realsense T265
- Intel Realsense D435i
- Webcam Logitech HD C270
- RasPi Cam IMX219-160 IR
- Camera holding support

Ubuntu 18.04 is installed on the PC since, from previous experience, it is the version with more compatible with the libraries and software used in this project. For the purposes of this research, it is chosen to use ROS2 which enables real-time, therefore is installed the Dashing Diademata version, compatible with this version of Ubuntu. Then several Python libraries are installed on the PC to allow OpenCV 4 (computer vision library) to work properly, which among other things is used for fiducial marker recognition. Finally, programs for managing and visualizing the Intel Realsense data are installed.

On the Jetson Nano, some compatibility problems occurred both during the installation of ROS2 and during the installation of OpenCV4 library. There are compatibility problems with Ubuntu 18.04 due to the different processor from the Asus PC; so after many unsuccessful attempts of troubleshooting in the end the solution was to change Ubuntu version and install Ubuntu 20.04.5 LTS. With this operating system, it was also necessary to change the version of ROS2 and upgrade to Foxy Fitzroy. No Intel programs were installed on the Jetson as Intel cameras would be used via PC. The Jetson Nano was required to do testing with the Raspi Cam, because it does not have the possibility to be connected via standard USB, and moreover to simulate nodes' performance on a realistic embedded computer. In fact, a standalone drone, cannot be designed to be controlled remotely via a PC (which would introduce huge latencies and other problems) but the intelligence must be on board the drone, and the only way to do this is to equip the drone itself with an on-board computer that is light enough to allow it to fly (without oversizing the drone), is performant (so that it can compute even complicated algorithms such as EKF or pose estimation), and is power efficient (in order to have an acceptable flight range without the need for very large batteries, which would raise cost and overall weight)

Once ROS2 and OpenCV4 were installed on both machines, namespace configuration was done and the environment was prepared for development and testing of the various nodes. For the purpose of their use, the 2 versions of ROS2 Dashing and Foxy were interchangeable and perfectly compatible.

4.2.1 What is ROS?

ROS (Robot Operating System), serves as an open-source middleware platform, facilitating the development, control, and coordination of a wide range of robotic applications. Contrary to its name, ROS is not a traditional operating system; instead, it operates as a flexible and modular framework that functions on top of the existing OS. It offers a comprehensive suite of tools, libraries, and conventions that empower engineers, researchers, and developers to efficiently design, control, and orchestrate robotic systems. It acts as a communication and coordination focal point, facilitating the seamless exchange of data, the control of hardware, and the execution of intricate tasks in a modular and extraordinarily adaptable manner.

A key feature of ROS is its node-based architecture. In this architecture, a "node" represents a fundamental computational unit, akin to a software module or component. Nodes can be created in a variety of programming languages and can be executed on different computational platforms. These nodes are self-contained programs that execute specific tasks or functions and communicate with each other using a publish-subscribe mechanism, which is crucial for enabling real-time communication among nodes. This model fosters loose coupling, allowing nodes to work independently and ensuring that changes to one node do not adversely affect the operation of other nodes. Moreover, the publish-subscribe model supports both one-to-many and many-to-one communication, which enables distributed, modular, reusable, and scalable robotic system design.

Publisher Node Publisher nodes are responsible for transmitting data, often referred to as messages, to the ROS network. These nodes typically collect and publish data related to specific sensor readings, camera images, sensor measurements, or the current state of a robot. The publish-subscribe mechanism disseminates data to the ROS network, where multiple subscriber nodes can access and use this data. For instance, a laser scanner driver node is a common example of a publisher node, which publishes laser scan data to the ROS network for further processing.

Subscriber Node Subscriber nodes, in contrast, are designed to receive data published by publisher nodes. These nodes subscribe to specific topics, which represent particular types of data, to obtain the information they require. Upon receiving data, subscriber nodes process this information and carry out various tasks based on the data's content. For instance, a navigation planning node might subscribe to laser scan data to detect obstacles and plan a robot's path accordingly.

ROS vs ROS2 In 2014, the development of ROS2 began as a significant evolution of the original ROS. Although ROS was largely used for research and development in robotics for industrial applications that were gradually becoming widespread, it had some limitations that led to the creation of ROS2. The main distinctions between the two are:

- Communication: ROS relies on the use of a middleware called the Robot Operating System Framework (ROS-Framework) to facilitate communication between various components (nodes). ROS2, on the other hand, leverages the Data Distribution Service (DDS) standard for communication, which provides improved support for real-time and distributed systems, making it more suitable for commercial and industrial applications.

- Real-time Support: ROS2 incorporates native real-time support, which allows it to better handle time-sensitive tasks, a critical requirement in many robotic applications.

- Platform Independence: ROS2 is designed to be platform-independent, making it more versatile and capable of running on a broader range of hardware and operating systems, including real-time and embedded platforms.

- Improved Security: ROS2 places a stronger emphasis on security, offering features like authentication and encryption for communication between nodes.

- Language Support: ROS2 extends its support to additional programming languages, including C++, Python, and Java, making it more accessible to a wider audience.

4.3 Phase 1: Sensor fusion, VIO and fiducial marker relocation

4.3.1 First nodes implementation

With ROS it is possible to implement, as we have seen, in addition to several constructs, publisher, and subscriber nodes. With publisher nodes is possible to publish streams of sensor data: in our case, the odometry data of the VIO pose computed by the IntelRealsense T265 camera, and the pose data estimated as a result of fiducial marker identification. As a first step, thus, was developed a publisher node for the webcam ("webcam_image_publisher" that would publish image streams. The corresponding publisher node for the T265 camera was not developed because the camera itself, in addition to publishing VIO odometry data (see an example in fig [4.4]), it publishes also video streams of the 2 optics plus other information/topics.

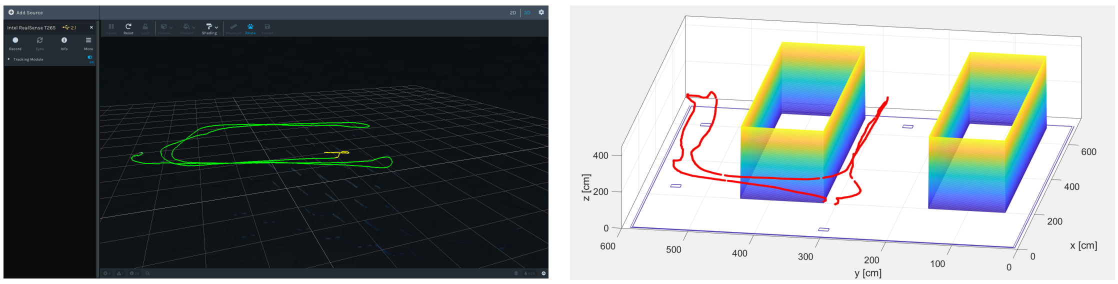


Figure 4.4: VIO odometry from Intel Realsense T265 and the same data plotted on Matlab.

Therefore, a listener node was developed that subscribes to the publisher node of the respective image stream and enables image saving. To ensure that the pose estimation obtained from the fiducial markers is accurate and reliable, it is indeed necessary to calibrate every single camera, and then use the obtained distortion matrix to make the pose estimation computation. This tricky but likewise important issue will be discussed in a forthcoming section (see section 4.4.3). After having calibrated the webcam, therefore, it was developed the "ArUco_publisher" node, which is both a listener and a publisher at the same time. This node subscribes to the image stream publisher node, receives the images, identifies all the markers (ArUco markers were used in this first phase) present in the single frame, returns the coordinates of the ArUco, and then the pose is computed using the Rodriguez formula. Finally, the obtained pose estimation data are published.

During the development of "ArUco_publisher" node, it was noticed that the frequency at which this data was being published was very low, around 7.5 Hz, which is not a sufficient frequency for a real-time application in this domain. Since this node was developed in Python, which doesn't make performance its strong point, therefore, it was decided to try to increase the performance of the algorithm by remaking the node in C++. Despite this, however, the frequency at which the node was publishing pose data was almost unchanged, which meant that the bottleneck was not the performance of the algorithm written in Python but of the camera itself. Seeing that the performance of the 2 nodes written in Python and C++, for this specific algorithm, were roughly identical, since for this type of computer vision application, which leverages OpenCV and other libraries that are very easy to integrate and use in Python, it was decided to develop all further nodes in Python. A specific mapping between ArUco IDs and offset coordinates was done: in fact, 4 different ArUco were placed, in 4 different locations, each time the webcam identified a specific ArUco, the corresponding X and Y offset were applied to the estimated coordinates to effectively translate.

4.3.2 Base sensor fusion node

The last step was to implement a listener and publisher node ("fuse_sensor") that would do the sensor fusion operation. The goal of this first phase is not to implement complicated logic, advanced Kalman filters, etc., but only to get all the data to stream properly and do a preliminary test. So this sensor fusion node basically subscribes to the VIO publisher node that publishes the pose odometry data, and to the "ArUco_publisher" that publishes the pose estimation, when these are present and visible from the webcam.

The first implementation with the basic logic was structured as follows:

1. Odometry data received from VIO
2. A translation is performed with respect to the coordinates of the true starting point: the VIO when turned on starts from [0,0,0], but in our test environment the initial coordinates are different from [0,0,0].
3. These new resulting coordinates, which are published, will exactly match the coordinates received from the VIO, except for the initial translation, as long as no ArUco is detected and thus no relocation is performed.
4. When an ArUco marker is detected, the coordinates received from the camera pose estimate with respect to that specific ArUco are summed and effectively merged with the original VIO coordinates. Complex filters and logic should be applied at this stage, but that was not the purpose of this phase of the project. With this fusion operation, a relocation was effectively performed,

and "theoretically" the accumulated drift error from the VIO odometry should be reset.

4.3.3 Preliminary results: fiducial marker relocation issue

Below, in 4.5, 4.6 and 4.7 you can find a plot with the 3 coordinates plotted. The red line represents the original VIO, as you can see it is actually very accurate in our environment, thanks also to the fact that the environment in which the tests were carried out was well-illuminated and rich in features to be extracted. The blue line, on the other hand, represents the "fused" coordinates, i.e., those published as output from the "fuse-sensor" node, which was translated when the tracking camera was turned on and then relocated whenever an ArUco was detected. As can be seen, following the relocation, these coordinates become very imprecise and lose their meaning. This is certainly due to the used logic to do the relocation (no filters but bare sum); but beyond this factor, we can still observe that the point cloud in green, which represents the pose estimate calculated through the ArUco (blue square on the ground), is very noisy and wide and therefore inaccurate, thus making very difficult to apply any filter or relocation logic. Based on these results, it was decided to proceed with phase 2: to analyze the pose estimation by fiducial marker in more detail, and then make an ad hoc comparison.

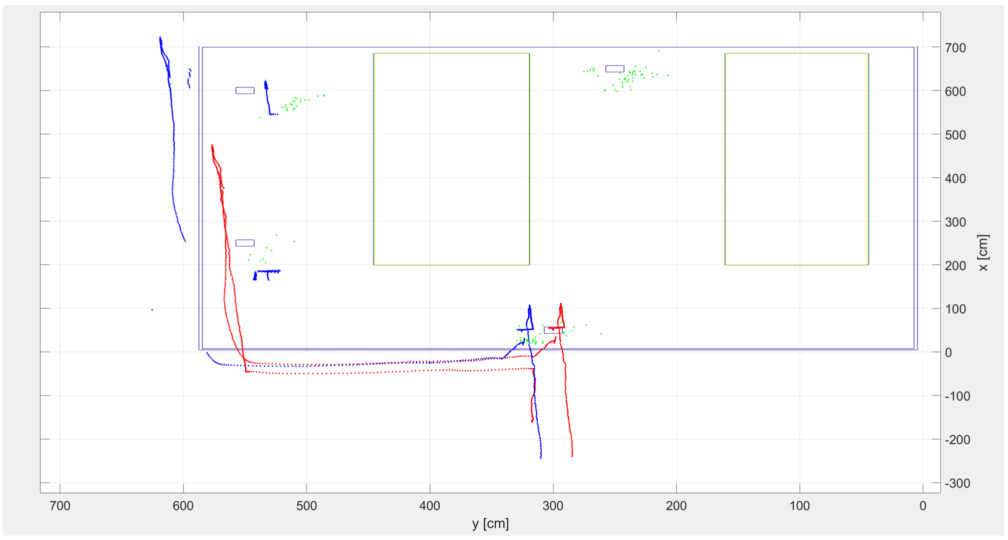


Figure 4.5: Plot of fused odometry data: in red VIO, in green ArUco pose estimation and in blue fused data

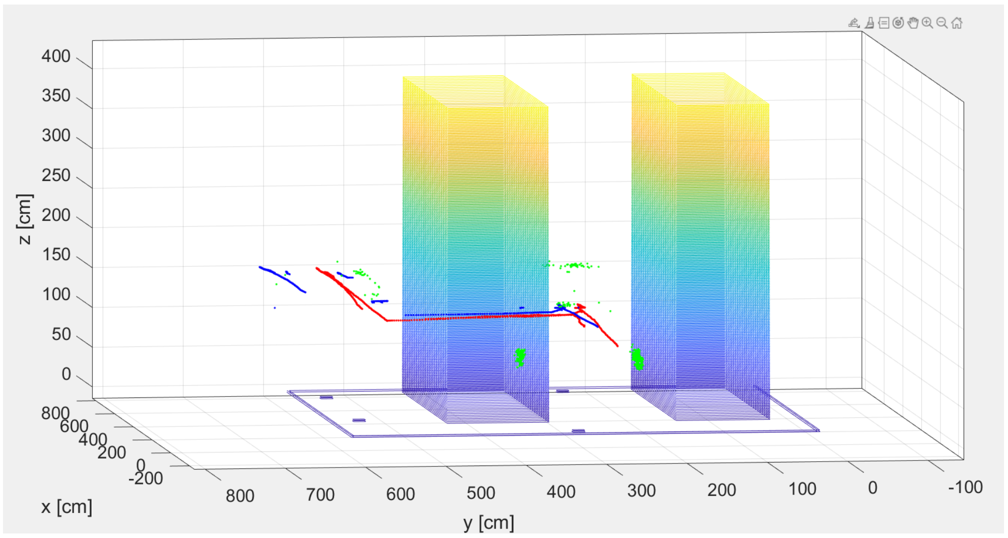


Figure 4.6: Plot of fused odometry data: in red VIO, in green ArUco pose estimation and in blue fused data

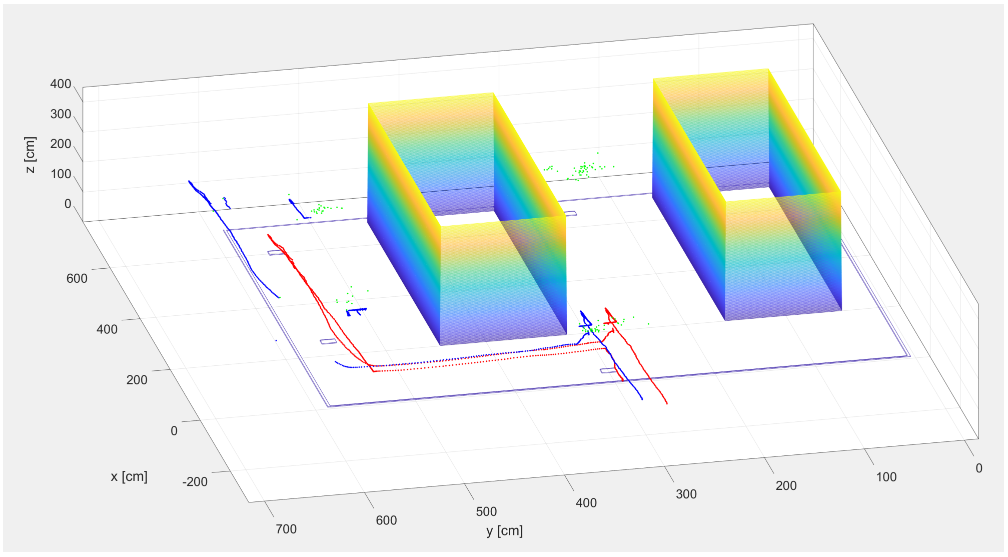


Figure 4.7: Plot of fused odometry data: in red VIO, in green ArUco pose estimation and in blue fused data

4.4 Phase 2: comparison ArUco vs AprilTag

As mentioned in the previous section, given the results obtained from sensor fusion it was decided to do a more extensive study on fiducial markers. Therefore, it was chosen to compare 2 of the most commonly used fiducial marker types for this type of systems: ArUco (used for example by Doks) and AprilTag (used for example by Boston Dynamic's Spot or Yape). Whereas, regarding cameras, preliminary research was done to understand which cameras are used today by systems of this type.

4.4.1 Type of used camera

Objective: identify which cameras are used (for fiducial marker recognition) by the main autonomous robots on the market so as to do tests on the same or similar cameras. After some research, the results were not very detailed and not very helpful since because they are commercial products, manufacturers tend not to provide details of the used hardware, but only some features (not too specific) and what kind of operations or algorithms they are able to perform with them. Below are some features of the main cameras.

- Boston Dynamic – Spot
  - 5 stereo pairs cameras (front-left, front-right, left, right, back) for a 360 degrees field of view
  - Black-and-white or color fisheye, range (depth), infrared
  - Used for robot perception (also using Aruco marker) and obstacle avoidance
  - Depth camera range ≈ 2 m
- Xiaomi – Cyberdog
  - Intel RealSense D450 depth camera
- General Laser – Unitree Go1
  - 360 degrees field of view
  - 5 Sets Fish-eye Stereo Depth Cameras (Ai Post-processing)
  - Lens Angle ≈ 150° x 170°
- General Laser – Unitree Go1
  - Active infrared stereo depth technology
  - 1080P camera resolution
  - Depth distance 0,3 - 10m
  - Error accuracy within 2 meters is less than 2%
- Anybotics – Anymal

  - 5 stereo pairs cameras for a 360 degrees field of view
  - No information. It seems to be an Intel RealSense fisheye stereo depth camera (like the others)

- Taurob
  - 360° degrees camera, not specified how. Not sure uses Aruco

As mentioned above the models of the specific cameras used were not found then (I think some of these robots in fact even use custom, home-built hardware es Boston Dynamics). However, it can be seen that most of these systems use a set of stereo depth cameras with features similar to the Intel RealSense D-series (if not even a specifically stated model).

Based on this research and the availability we had in the laboratory, the following cameras were chosen:

- Intel RealSense - T265
  - Stereo Tracking camera, already used for VO, with IMU
  - Two OV9282 Fisheye Camera, a monochrome image sensor with a wide field of view
  - Fisheye FOV with Cover Window: 170°
  - The distance between the images, which is referred to as the baseline or intraocular spacing: 64 mm
  - Focus: fixed
- Intel RealSense - D435i
  - Stereoscopic RGB- Depth camera with inertial motion unit
  - Working range 0,3 m to 4/5 m
  - Depth Field of View (FOV): 87°(H) x 58°(V) x 95°(D)
  - Depth output resolution: Up to 1280 x 720
  - Depth frame rate: Up to 90 fps
  - RGB max frame resolution: 1920 $\times$ 1080 (it is possible, eventually to use a reduced resolution set)
  - RGB sensor FOV (H $\times$ V x D): 69° x 42° x 77°
  - RGB sensor resolution: 2 MP
  - RGB frame rate: 30 fps
  - IR-pass filter

  - Wide Infrared Projector: H:90 / V:63 / D:99
  - Baseline: 50mm

- Logitech webcam HD C270
  - Frame resolution 1280 x 720
  - Resolution 0.9 Megapixels
  - Fixed Focus
  - Sensor FOV 55°
  - Frame rate 30 fps

- RasPi Cam IMX219-160 IR
  - Frame resolution: 3280 × 2464
  - Resolution: 8 Megapixels
  - Sensor: Sony IMX219
  - Supports night vision when working with infrared LEDs
  - Sensor FOV: 160°
  - Lens specifications: CMOS size: 1/4inch
  - Aperture (F): 2.35
  - Focal Length: 3.15mm
  - Distortion: < 14.3%
  - Frame rate 30 fps
  - Focusing Adjustable Range: 20cm - infinity (Depth of field)

Since the purpose of this research is on drones and systems capable of autonomous navigation, particular attention was paid to certain characteristics and features in choosing the cameras. Indeed, all these cameras have in common the fact that they are extremely light, small, and have low power consumption. In addition, they are generally quite high-performance, apart from the Logitech, which was purposely chosen low-end to have a well-rounded overview.

4.4.2 Fiducial Marker families: why ArUco Original and AprilTag 36h11?

Generally speaking, depending on the fiducial marker's bit size, there are different levels of precision and unique identification in image processing. More bits in a marker's code dictionary reduce the likelihood of confusion but require higher resolution for accurate detection. Conversely, smaller bit sizes aid marker identification with small or distant images but limit the total unique markers available. Additionally having a lower amount of markers decreases the inter marker distance, thus the chance of faulty marker ID classification.

AprilTag AprilTags are two-dimensional bar codes (similar to QR codes) designed to encode smaller data payloads (between 4 and 12 bits), allowing them to be detected more robustly and from longer ranges. They are also designed for high localization accuracy, allowing for precise 3D position computation with respect to the camera. There are two generations of AprilTag families: Apriltag 2 and AprilTag 3. Apriltag 2 is the classic Apriltag, compatible with more software and offering a better out-of-the-box experience. Apriltag 3 is the newest generation, offering new features like circle tags, recursive tags, faster detections, and higher recall rates but less software support.

AprilTag libraries families are characterized by two numbers: the first number (before h) represents data bits (changeable blocks) in the tag design, and the second (after h) is the Hamming distance, the minimum number of bits that must be changed in one tag's code to reach another tag's code. More bits mean more tag IDs are available in that family, and a larger hamming distance allows for more error correction while decoding tag IDs. However, a larger hamming distance reduces the number of available tag IDs, such as 36h11 having more available IDs than 36h15 but a higher false positive rate.

The 36H11 family is a reasonable generic choice due to its even size and high fill factor, but its larger width may limit the range. The more bits a tag has, the more pixels are required to decode it, allowing lower-bit tags to be detected further away. In conclusion, AprilTag gen 2, specifically the family 36h11, is recommended for most applications due to its robustness and the availability of most tag IDs.

ArUco An ArUco (Augmented Reality University of Cordoba) is an open-source library designed for generating, detecting and recognizing squared fiducial markers in images. These markers are generated based on specific criteria to maximize inter-marker distance and the number of bit transitions, ensuring their distinctiveness and thus reducing the likelihood of confusion during detection. ArUco offers the unique advantage of allowing users to create configurable libraries, tailored to their specific needs, rather than relying on a standard library with a fixed set of markers. These libraries will contain only the specified number of markers with the greatest Hamming distance. This customization reduces computing time and enhances detection accuracy [25].

The ArUco library provides 25 predefined dictionaries of markers, each containing markers with a fixed number of blocks (or bits) and a specific number of unique IDs. It is chosen to use the ArUco "original" dictionary because it strikes a balance between marker size and detection accuracy, making it suitable for a wide range of applications such as augmented reality, computer vision applications, and pose estimation. Moreover, many autonomous solutions on the market already use this specific dictionary, such as Doks, so even for this reason it is decided to this specific dictionary. Finally, Aruco is compatible with OpenCV: in fact, it is possible to generate custom markers and estimate their position (as will be presented in a further section) using specific OpenCV functions.

4.4.3 Camera calibration

Camera calibration is an extremely important operation and should therefore be done with extreme care and precision. Each camera and in particular each individual lens (even for the same camera model) has specific and unique characteristics, so although to the human eye, 2 captured images from the same camera model can appear to be exactly the same, in reality, there is a slight difference due to the distortion that each individual lens brings (that as we will see characterize the camera matrix). Let's examine this phenomenon more in detail. As explained in [26] it uses a pinhole camera model to project a scene's 3D point P_w into the image plane, forming the corresponding pixel p. Both P_w and p are represented in homogeneous coordinates, 3D and 2D homogeneous vectors respectively. The distortion-free projective transformation given by a pinhole camera model is shown below:

$$s\, p = a[R|t]P_w$$

where P_w is a 3D point expressed to the world coordinate system, p is a 2D pixel in the image plane, A is the camera intrinsic matrix, R and t are the rotation and translation and s is the projective transformation's arbitrary scaling and not part of the camera model. The camera intrinsic matrix A is composed of the focal lengths fx and fy, which are expressed in pixel units, and the principal point (cx,cy), that is usually close to the image center:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

The intrinsic parameters of a camera, such as the focal length of a zoom lens, can be re-used once estimated, provided the focal length remains fixed. However, if an image is scaled by a factor, all parameters must be multiplied or divided by the same factor. From the projective transformation that maps 3D points in world coordinates into 2D points in the image plane and in normalized camera coordinates we obtain:

$$s\begin{bmatrix}u\\v\\1\end{bmatrix}=\begin{bmatrix}f_x&0&c_x\\0&f_y&c_y\\0&0&1\end{bmatrix}\begin{bmatrix}r_{11}&r_{12}&r_{13}&t_x\\r_{21}&r_{22}&r_{23}&t_y\\r_{31}&r_{32}&r_{33}&t_z\end{bmatrix}\begin{bmatrix}X_c\\Y_c\\Z_c\end{bmatrix}$$

and if $Z_c \neq 0$

$$\begin{bmatrix}u\\v\end{bmatrix}=\begin{bmatrix}\frac{f_xX_c}{Z_c}+c_x\\\frac{f_yY_c}{Z_c}+c_y\end{bmatrix}$$

with

$$\begin{bmatrix}X_c\\Y_c\\Z_c\end{bmatrix}=[R|t]\begin{bmatrix}X_w\\Y_w\\Z_w\\1\end{bmatrix}$$

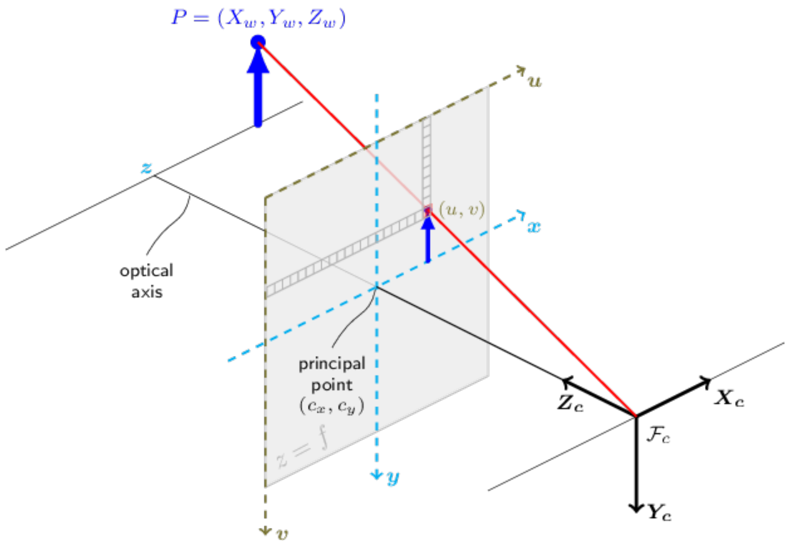


Figure 4.8: Pinhole camera model

Pinhole cameras can cause significant distortion in images, primarily radial and tangential distortion. The pinhole model is extended as:

$$\begin{bmatrix}u\\v\end{bmatrix}=\begin{bmatrix}f_xx''+c_x\\f_yy''+c_y\end{bmatrix}$$

Radial distortion makes straight lines appear curved, and its size increases as points move away from the image center. Radial distortion is represented as: [27]:

$$x_{Radial\ distorted}=x(1+k_1r^2+k_2r^4+k_3r^6)$$
$$y_{Radial\ distorted}=y(1+k_1r^2+k_2r^4+k_3r^6)$$

Tangential distortion occurs when the image-taking lens is not perfectly aligned with the imaging plane, causing some areas in the image to appear closer than expected. Tangential distortion is represented as:

$$x_{Tangetial\ distorted} = x + [2p_1xy + p_2(r^2 + 2x^2)]$$
$$y_{Tangetial\ distorted} = y[p_1(r^2 + 2y^2) + 2p_2xy]$$

with

$$r^2 = x^2 + y^2$$

So the distortion coefficients are $k1,\ k2,\ p1,\ p2\ and\ k3$,

Finally, other information, like the intrinsic and extrinsic parameters of the camera is also very important to complete the calibration. Intrinsic parameters, such as focal length (f_x, f_y) and optical centers (c_x, c_y), are crucial for analyzing camera performance. Extrinsic parameters are rotation and translation vectors that convert 3D point coordinates into a coordinate system. These parameters, unique to a camera, can be used to create a camera matrix that removes distortion caused by the lenses. The focal length and optical centers can be expressed as a 3x3 matrix, making it a useful tool for analyzing images taken by the same camera.

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

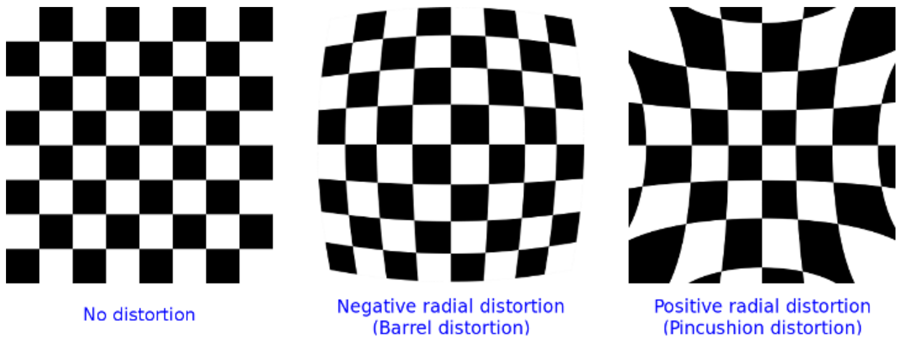


Figure 4.9: Example of distortion, image from [26]

Applied procedure to calibrate cameras

To calibrate cameras, it was used a 9x6 chessboard and a Python algorithm exploiting some specific functions of the OpenCV library provide as output the camera matrix and the camera distortion: the 2 matrices that characterize the

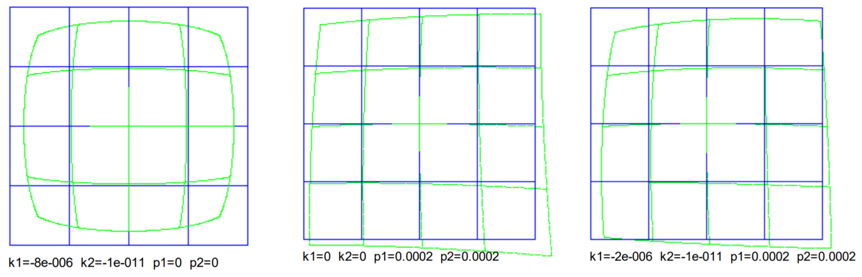

k1=-8e-006 k2=-1e-011 p1=0 p2=0
k1=0 k2=0 p1=0.0002 p2=0.0002
k1=-2e-006 k2=-1e-011 p1=0.0002 p2=0.0002

Figure 4.10: Example of tangential and radial distortion, image from [28]

specific camera lens. These matrices are then loaded at node startup to correct the distortion of the specific camera, and thus obtain an undistorted image resulting in better pose estimation results. The procedure itself is very simple: it consists of taking multiple photos of the chessboard at different orientations, angles, and distances (see Fig. 4.11. The more different photos you can take, the better the

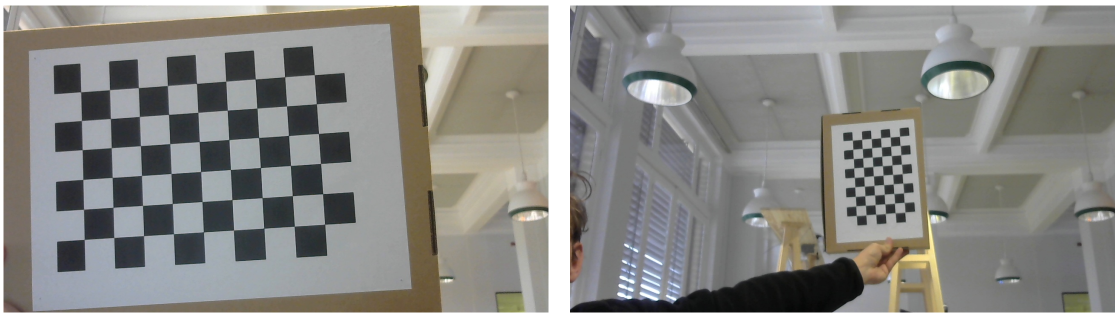


Figure 4.11: Example of calibration chessboard image acquisition

resulting calibration will be. Once all the photos have been collected, the algorithm that calculates the 2 matrices is launched: the algorithm analyzes one photo at a time and tries to extract and recognize the corners of the chessboard squares; this is the used function:

```
    ret, corners = cv2.findChessboardCorners(gray, (nCols,nRows),
   None)
```

and this is the visual obtained output (fig. 4.12).

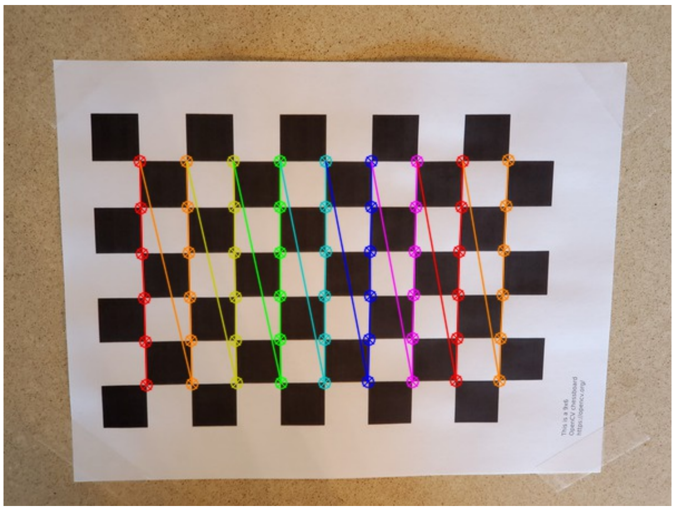


Figure 4.12: Example of corner recognition on a calibration chessboard

Once all the corners of the squares in the chessboard have been collected, they are saved in a data structure and used after having analyzed all the images, to compute and generate the matrices.

```
    ret, mtx, dist, rvecs, tvecs = cv2.calibrateCamera(objpoints,
   imgpoints, gray.shape[::-1], None, None)
```

where *mtx* e *dist* are respectively the calibration matrix and distortion matrix. All photo acquisitions were made under the same illumination conditions that were good and uniform so that the calibrations were not affected by bad illumination conditions. Once the calibration matrices were obtained, an error was calculated by considering the calibration values just obtained. Not having a robotic arm with which to set all the points from which to take pictures of the chessboard, this operation (movement of the chessboard in 3D space) was carried out manually: therefore 2 calibrations of the same camera, performed with different sets of pictures gives slightly different results. To obtain the best possible calibration, the whole procedure was repeated several times, taking different photos, and then selecting the best calibration. It was noted that more photos do not necessarily imply a better calibration, and vice versa after a certain number of photos, the error increases instead of decreasing. Therefore, several calibrations were made considering more or fewer photos, to obtain the result with the lower error, this calibration then was selected as the default one for subsequent measurements. It should be noted that at this stage extreme care was taken as a small error at the calibration level would have resulted in biases in the measurements that would have been difficult to interpret and correct. Data from the calibrations and their errors are collected in the following table.

Calibration data		
Camera	Pictures number	Error
Intel D435i	31	0,087
Raspi Cam	22	0,052
Logitech Webcam	26	0,040
Intel T256 Lens 1	24	0,134
Intel T256 Lens 2	20	0,168

As can be seen, the 2 fisheye optics of the T265 camera have a very high error, because as seen, the images that are obtained from this type of optics are very distorted and therefore it is very difficult to calibrate them properly and correct a relevant distortion. While regarding the number of used pictures there is no specific number, but from the tests conducted some calibrations gave better results with fewer pictures and others with more pictures, in any case, an average of 20 to 30 pictures were used per calibration.

4.4.4 ROS node implementation and adaption

Starting with the "ArUco_Publisher" node previously developed, the other nodes were implemented to recognize ArUco and AprilTag using different selected cameras. The core of the nodes basically remains the same: only some parts were modified allowing subscription to the different video streams published by each camera. In some cases, it was also necessary to adapt the frame management, for example for the 2 fisheye optics of the T265, which because they are monochrome (grayscale) require different frame processing. For this camera tests and measurements were made on both optics, in order to verify that the results between the 2 optics were consistent and coherent with each other.

On the other hand, regarding the fiducial marker recognition component, the effort was directed to keep the algorithm as similar as possible (between the version that recognizes ArUco and the other one that recognizes AprilTags), in order to limit the variables involved for the subsequent measurement and testing phase and thus to facilitate analysis. Hence, only the library used to recognize fiducial markers was modified: in both cases, the resulting output upon recognition is an array containing the ID and the 4 coordinates of the 4 vertices of the marker together with other information. The pose estimation is done following these steps:

1. Four marker vertex are recognized and their coordinates are saved: to do this, specific functions of the libraries ArUco and AprilTag were used

2. By using the function

```
    aruco.estimatePoseSingleMarkers(corners, marker_size,
   camera_matrix, camera_distortion)
```

of the ArUco library, which is used for both fiducials in order to reduce variability, the rotation and translation vectors are obtained. It is seen that in the used function both the marker size and the camera characteristics are taken into account: the camera matrix and the camera distortion matrix, which were obtained during calibration

3. Having obtained the rotation and translation vectors, it is possible to calculate the pose using Rodriguez's formula with the computations seen in the previous chapter

4. Finally, the data stream, in addition to being published by the node, is also saved in a file, so that when all the measurements are finished, they can be processed through a Matlab script in order to compute the mean on X, Y and Z, the standard deviation on X,Y and Z, and the absolute and percentage deviance (error) of the reference axis (X, Y or Z)

Once all the nodes were developed and the first tests were performed for the selected cameras, it was confirmed again that the "bottleneck" and the low ratio in publishing the poses for the Logitech webcam was due to the poor performance of the webcam itself; in fact, with the 2 Intel cameras, equipped with an internal processor, the data flow had a frequency around 25 HZ, that it is speedy and performing. This enabled the exclusion of possible inaccuracies in pose estimation which can be attributed to too high computation problems and/or limited processor performance.

Note 1: it is important to specify that for the Intel D435i camera, only the RGB sensor was used for this type of measurement; the depth sensor was not used.

Note 2: the webcam and the 2 Intel cameras were plugged into the PC, and the tests were done by running the ROS nodes on the PC, while for the Raspi cam, not having the USB connection was used the Nvidia Jetson Nano board. This choice was made for 2 reasons: the first one was due to the installation of the Intel Realsense packages that gave some problems on the Jetson and being a very demanding process already done on the PC, we did not want to waste time reinstalling everything also on the board, the second one instead was the intention to parallelize and speed up all the measurements done since many measurements had to be done that would have taken a lot of time (even doing so they still took a lot of time). However, this choice was validated as several runs of measurements were made with the webcam, with the same lighting conditions and distance from the fiducial markers to verify that the results obtained by running ROS on the

PC and on the Jetson were repeatable and comparable. Having obtained almost identical results, it was possible to validate the aforementioned choice and utilize this setup for measurements.

4.4.5 General consideration for fiducial marker sizing and test approach

For the first set of measurements, I decided to consider an Aruco marker with a size of 13.8 cm per side assuming that the reading distance between the room and the fiducial marker would be just a few cm, up to one meter. I made this assumption based on the fact that the width of warehouse aisles is 180 cm on average. Considering that the drone flies in the middle of the shelf, the distance between the camera and the rack facade is about 60/70 cm, assuming that the drone is about 40/50 cm wide and the cameras are installed on the drone's sides. This is the distance that has been assumed with respect to the markers installed on the rack facade that help the system relocation within the aisle (even at considerable heights). So considering standard aisle dimensions, the reading distance will always be about 70 cm, as the aisle width remains constant. It should be noted that these considerations were made because the drone's side camera, in addition to performing relocation by fiducial marker, is also the camera that would scan the barcodes or QR codes of the goods to be inventoried within the warehouse during the stock take operation. By fixing the markers on the shelf and then doing the relocation in a perpendicular plane to the floor, there is the great advantage of keeping high accuracy in the Z axis even though the drone is at very high altitudes. In this case, the accuracy on X-Z or Y-Z plane (depending on the coordinate of reference system orientation with respect to warehouse corridors direction) would be very high. We will come back to this because this is a very important aspect

Regarding relocation outside the corridors, in moving from one corridor to another instead, it was considered that the drone could fly at a low altitude between 80/100 cm to 150/200 cm. In this way, the markers could be placed on the floor so that they could be easily detected by a camera pointing downward, even without walls or shelves to attach the markers to. It was designed to fly the drone at "human height" to avoid flying it too high in order to minimize damages in case of a crash during these displacements; on the other hand, the drone cannot be flown too close to the ground for 2 reasons: firstly, the camera's view field pointing downward would be extremely limited, and secondly while flying drone propellers generate turbulence that affects the drone's flight dynamics, and thus creating a lot of vibrations prevent the T265 (VIO) to work properly.

The limitation of having markers of about 14 cm is that relocalization on the X-Y plane might be done only at low altitudes (about 1/2 m) while with the drone at high altitudes, it would be possible to rely just on the markers placed on the shelf face. In fact, by the time it moves even a few meters away (>200cm), the calculated position becomes very inaccurate and the error very large, therefore the measurement is basically ineffable: this is because the camera has difficulty in recognizing the marker, as the marker is very small and the resolution of the camera is not optimal (especially for the webcam).

To relocate the drone on the X-Y plane even at more considerable altitudes, it is necessary to increase the dimensions of the marker to make it detectable and clearly visible even at larger distances. Thus, it was decided to make a set of measurements by doubling the side dimensions of the marker, i.e., to use a marker of 28 cm side. By doing so, the range of working distance is expected to increase: the distance at which the estimated pose measurements are still reliable and with limited and acceptable error.

Taking these measurements and analyzing them will provide a basis for understanding whether there is any relationship between the marker size and the accuracy of the calculated poses in relation to the distance. It will also be possible to benchmark the measurements by comparing the rate of resolution between the camera and the size of the marker. In order to have a more complete picture it was thought to replicate the same measurements also with another type of marker used very often in the field of automatic robotics: after several researches, it was thus chosen to use AprilTag type as a comparison to ArUco because of its characteristics.

Summarizing the set of measurements under study is:

- ArUco of 13.8 cm side
- ArUco of 28 cm side
- AprilTag of 13.8 cm side
- AprilTag of 28 cm side

4.4.6 Measurements setup and execution

A dedicated area in the laboratory was set up to do all these tests and measurements. The 2 fiducial markers were placed on the wall at the same height and at a known and fixed distance between them. This allowed the 2 marker measurements to be made in parallel (to speed up the process), thereafter this offset among the 2 markers was subtracted during data saving. To make the measurements then, the procedure is straightforward. The test-bench with the cameras is placed at a specific known position in space, and the nodes are launched for a few seconds to start the measurements, so that the pose estimates are saved for later analysis. For each pose, about 400 measurements i.e., 400 coordinates (X,Y and Z) were recorded. Obviously, for the 2 Intel cameras, it took a few seconds to acquire all this data, while for the webcam and Picam, it took a little longer. It was decided to take several measurements to be sure to minimize and make negligible any spurious reading, as there was no data filter. Once the measurement at a given position is finished, the test bench is shifted by a known delta, the new position is noted, and the next measurement is performed.

To maintain order and methodology in the measurements, only one dimension is varied at a time. For both small fiducial markers, for each camera, five sets of measurements were made: one set of measurements on the Z axis, two sets of measurements on the X and Y axes respectively at a closer Z distance (between camera and marker), and two more sets of measurements at a more distant Z distance. For the bigger fiducials, on the other hand, only one Z distance was chosen at which to make measurements for the X and Y axes.

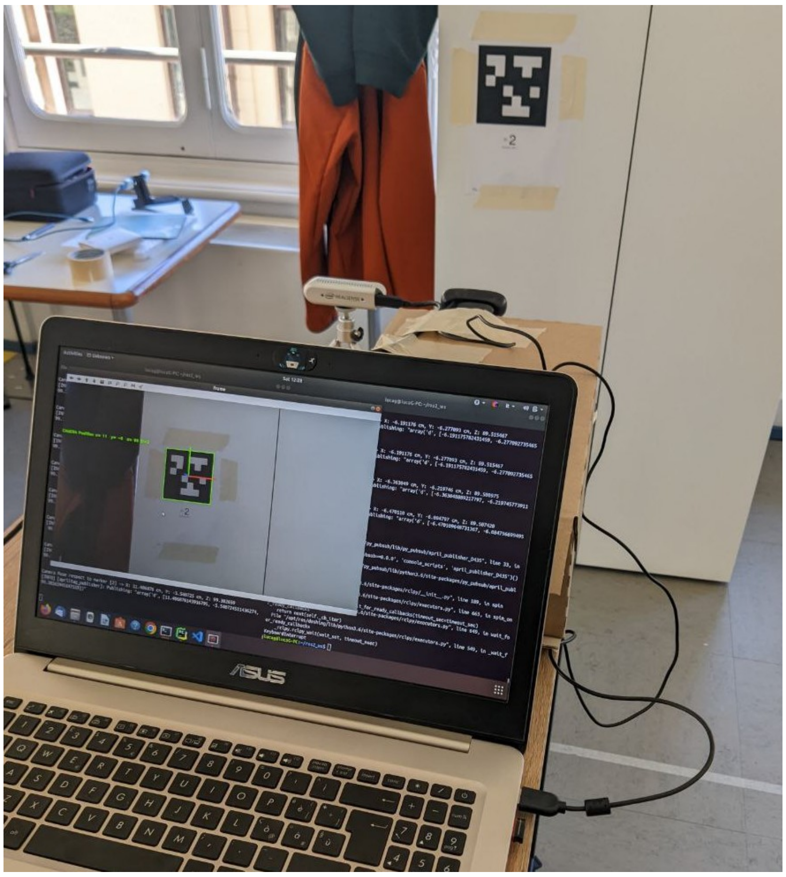


Figure 4.13: Example of a measurement session

Z-axis measurements: The first step is the measurements on this axis, in order afterward to choose the 2 distances for the subsequent sets of measurements based also on the results obtained in this first set. Regarding the Z-axis measurements, it was avoided to position the camera exactly in (X: 0; Y:0) in order to prevent the data from being subject to Z-flipping (that will be analyzed later in Chapter 5), and thus to obtain the most accurate and cleanest data possible. So the procedure involves starting the measurements by placing the test bench in an initial position, and taking the measurements with all the cameras and then moving the test bench further from the markers, moving it along the Z-axis by a known and fixed interval.

With this set of measurements, the purpose was to test whether and how much the accuracy of the measurements would change with varying the distance between the fiducial marker and the camera. Once the Z-axis measurements were finished for both fiducial markers, for all cameras, an initial analysis was done to decide at which distances, one closer and one farther, to make the measurements for the X and Y axis. The two distances were chosen by examining at which distance the best results had been obtained but also in combination with the considerations made in the initial phase of this fiducial marker study: i.e., based on the working range in of the small markers (approximately 60 to 200 cm). Considering both of these aspects, the distance of 70 cm was chosen as the closest distance because the results were good and it is a realistic distance at which the drone could be in the process of identifying markers applied on the shelving overhead. In contrast, 160 cm was chosen as the farther distance (for the navigation scenario outside the corridors) because at this distance the results were still good for all the cameras: beyond this distance, as discussed later in the conclusions, some cameras were starting to give inaccurate pose estimates with an increasing error, making the estimated pose completely unreliable.

However, for the larger fiducial markers, a single Z distance was selected to perform measures: 450 cm. Since the idea of having a larger fiducial marker is to be able to do relocation even at higher elevations and longer distances, having obtained and analyzed the first results on the Z axis, the largest feasible distance is chosen where all the cameras had pose estimates are still acceptable in terms of accuracy. This distance is significantly less than half of the working range for cameras such as the D435i and the webcam, while for cameras such as the wide-angle Picam or the T265, it is about $\frac{1}{3}$ and $\frac{2}{3}$ of the maximum working range, respectively. So, the final chapter will be analyzed also how these wide-angle and fish-eye lenses suffer distances, making it very difficult to recognize markers even at short distances.

Having chosen the Z distances for which to make X and Y axis measurements, the remaining measurements are carried out for all markers (type and size) and for all cameras.

X-axis measurements: To carry out the X-axis measurements, the procedure is similar to that on the Z-axis, but in this case instead of moving the stall away by a fixed delta, we proceed by moving and translating the stall laterally on the X-axis parallel to the plane of the markers. For the small markers, it was decided to take measurements in both directions of the X-axis: positive and negative. In this way it was possible to verify that the obtained results were mutually coherent and mirrored with respect to the Z axis, assuming that the calibration was accurate. This aspect will be analyzed in the results. Regarding the large markers, however, since the required test area was larger and space was limited, having verified from the previous measurements (on the small markers) that there is symmetry between the two axis directions (positive and negative), measurements were carried out only on one side, knowing that these measurements could have been flipped to the other direction as well.

Y-axis measurements: the measurements on the Y-axis were made in the exact same way as the measurements on the X-axis, except that the camera was mounted on the test bench rotated by 90°: this was done to avoid the need to make measurements considering an axis perpendicular to the floor and thus having to move vertically (which would have been complicated and inaccurate). As expected, the measurements made on this axis were fewer in number, since, especially for non-widely angled lenses, the vertical resolution (on the Y-axis - height) of the sensor is less than the horizontal resolution (X-axis - width).

The last consideration is the chosen interval for the shifts on the various axes for different marker sizes. For the 13.8 cm sided markers, it was decided to make the measurements at 10 cm intervals, based on the expected working range. For the larger markers instead, desiring to push as far as possible, and considering the much larger working range, it was decided to use a larger interval of 50 cm.

Chapter 5

Results and Conclusion

5.1 Results

5.1.1 Phase 1

Let's analyze the obtained results in Phase 1 and delve into the details to understand why it was decided to further investigate the measurements in Phase 2. In the following figure, the red line represents the output of the pose calculated by the VIO, while the green line represents the pose estimated through the ArUco marker. The measurements were conducted for approximately 74 seconds, and the data were

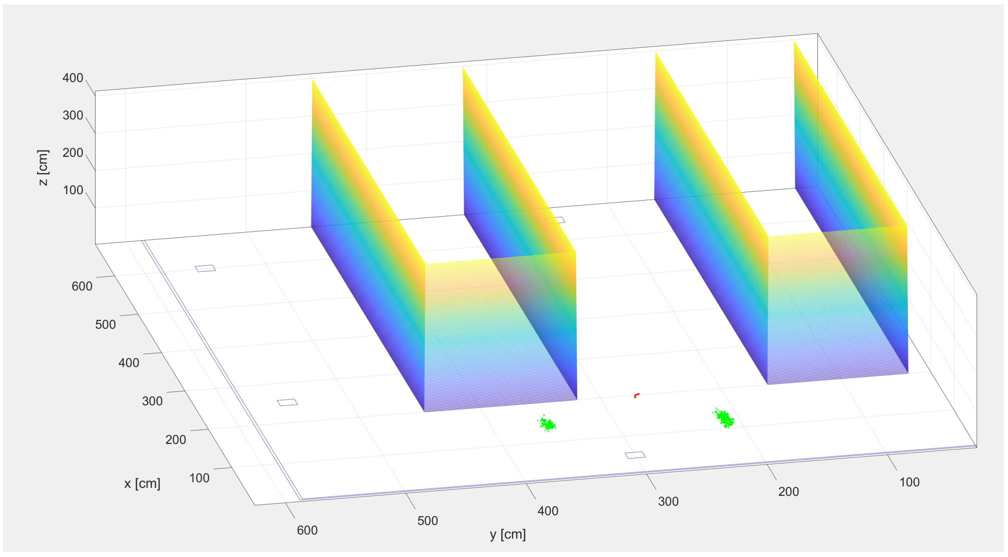


Figure 5.1: Plot of pose data of VIO - red and ArUco - green

published at a frequency of around 11 Hz. The starting point where the fiducial marker is located is at x = 50 and y = 300. As observed, the VIO measurements are highly precise over time, capturing even the slightest movement (as evident from the slight shift, due to the test bench being moved of few centimeters).

On the other hand, examining the green points, instead of being clustered around the red points near the perpendicular of the ArUco marker, they are grouped around two points, specifically at x = 50, y = 230, and x = 50, y = 370. These two clusters of points are symmetric with respect to the marker. This is not a coincidence but rather the result of a phenomenon known as z-flipping, which affects pose estimation with fiducial markers, especially within the ArUco family. This is a well-documented issue [29] [30], and while there are methods to limit and mitigate its effects, there is no definitive solution.

When the orientation suddenly changes from one image frame to the next, it should be possible to detect this and discard the spurious pose, since the problem must originate from an inaccuracy about the corner locations (also due to uncertainties about camera intrinsic parameters and distortion). Although filtering the poses is not a simple solution, one idea would be to apply an extended Kalman filter on quaternions, such as the Quaternion-based Extended Kalman Filter in order to mitigate spurious pose. It is also noted that this phenomenon occurs when the camera is exactly perpendicular to the marker or at the extremities of the camera's FOV. An alternative solution could involve implementing a threshold-based selection criterion for fiducial marker poses. Only poses falling within predefined ranges of distances and angles would be retained. This process would exclude fiducial marker poses if the distance exceeds a specified threshold or if the angle is close to 0° or significantly exceeds wide angles (about 100°), as poses in these conditions are highly inaccurate.

Given that the purpose of this research was not to delve into the details of the Z-flipping phenomenon and develop specific algorithms for its mitigation, therefore a comparative analysis was conducted to assess the precision of relocation using another fiducial marker and different cameras.

5.1.2 Phase 2

Frequency analysis

Let's analyze the frequencies at which the different nodes publish pose data streams.

Frequency of the stream pose data - Small Marker			
Camera	Distance [cm]	ArUco [fps]	AprilTag [fps]
Logitech Webcam	50	7,4	10,6
	100	7,5	8,8
	250	7,4	7,4
	700	5,8	8,6
Raspi Cam	50	11,6	13,9
	100	12,5	12,5
	250	9,7	12,8
	300	9,9	11,2
Intel D435i	50	25,0	25,0
	250	24,9	25,0
	300	23,5	25,1
	700	22,6	25,0
Intel T265 Lens 1	50	24,5	24,8
	200	23,1	24,7
	250	15,5	24,4
	280	18,2	14,8
	300	10,4	20,6
Intel T265 Lens 2	50	24,7	24,5
	200	24,7	24,2
	250	7,7	24,5
	280	-	17,6
	300	21,9	7,63

Frequency of the stream pose data - Large Marker			
Camera	Distance [cm]	ArUco [fps]	AprilTag [fps]
Logitech Webcam	100	14,1	25,1
	200	14,7	25,1
	300	10,6	25,0
	600	7,5	23,0
	1000	5,6	9,1
Raspi Cam	100	12,6	12,2
	200	11,3	12,2
	300	10,9	10,3
	600	9,1	12,1
	800	7,2	10,0
	1000	-	9,3
Intel D435i	100	25,0	25,1
	500	15,6	25,1
	700	23,6	25,0
	900	22,5	25,1
	1150	12,1	15,9
	1500	-	14,6
	2000	-	13,3
Intel T265 Lens 1	100	24,4	25,1
	300	23,4	24,9
	500	16,5	24,2
	600	2,17	23,8
Intel T265 Lens 2	100	24,5	25,1
	300	23,8	24,8
	500	16,8	23,8
	600	-	13,4

Firstly, it's important to note that the nodes were written to publish the data stream at the maximum possible frequency. Therefore, the different frequencies collected at various distances are indicative of the image quality captured by the sensor, considering the resolution, robustness, and reliability of the specific marker family. Another important point is that, for the two Intel RealSense cameras, the nodes publishing the image stream, to which the subscriber node subscribes to estimate the pose upon marker recognition, are generated by the cameras themselves. This means that the PC/board resources are not used in the image stream generation, therefore all resources are dedicated to the subscriber node for pose estimation. In contrast, for the other two cameras (Logitech and PiCam), computational resources are also responsible for generating the stream of images in the publishing node.

With these considerations, the performance in terms of frequency is significantly better for the AprilTag family. Using AprilTag markers, the publication frequency of poses is higher at the same distance. Furthermore, the maximum distance achieved with still acceptable publication frequencies favors AprilTags, which can be recognized quite well even at greater distances compared to ArUco markers. In some cases, the frequency of ArUco marker recognition is much lower than that of AprilTags, or for certain distances, ArUco markers were not recognized at all (e.g., 15m and 20m for D435i and 10m for PiCam). Nevertheless, in these cases, the frequency of pose estimation for AprilTags is quite high (approximately 14 Hz for D435i).

Comparing the two tables, it is observed that for Logitech and PiCam cameras, the frequency can reach up to 25 fps with larger markers at closer distances. Therefore, even these less performant cameras, when conditions allow for good marker recognition, can maintain an adequate frequency for real-time applications. However, as distances increase, the low image quality resulting from low resolution highlights the limited performance of these cameras. Indeed, as the distance increases, the frequency decreases drastically. The camera with the widest working range for both marker families is the D435i, maintaining 25 fps even at distances of 9m. Nevertheless, significantly better performance is observed for AprilTag markers in this case as well.

As expected, the performance of the T265 does not excel in terms of working range (max distance). Despite its high resolution, the optics of this camera, being fisheye, are greatly affected by distance. It has lower performance and a smaller working range along the Z-axis compared to all the other cameras analyzed. On the other hand, it has the widest field of view (FOV). Interestingly, the performance varies slightly between the two lenses, while one would expect them to be practically identical. For some distances, such as 280 cm, Lens 1 achieves a reasonably good frequency of 18,2 fps, while Lens 2 was unable to recognize the ArUco marker. This variation in behavior may be attributed to the difficulty in calibrating lenses that introduce significant distortion, such as fisheye lenses. A small difference in calibration can lead to significantly different behaviors in such use cases (for which the camera was not originally designed). It's worth noting that these two lenses reported a higher calibration error compared to all other cameras, and particularly Lens 2 had a higher error than Lens 1: Intel T265 Lens 1 error = 0,134 and Intel T265 Lens 2 = 0,168. This difference is also influenced by the fact that hardware may have defects, and thus, two theoretically identical sensors may have slightly different performances due to deformations or inaccuracies; they are not identical copies of software.

Pose estimation accuracy: Z-Axis

Let's now analyze how the accuracy of the estimated pose varies with changes in distance and deviation from the perpendicular. Not all measurements and graphs will be reported, but only the most significant ones that summarize and generalize the behavior of the two families of fiducial markers and different cameras.

We start by examining the absolute error and standard deviation of the measurements as the distance along the Z-axis varies. It's important to note that the absolute error was calculated by taking the difference between the mean value of all Z-coordinate measurements for a specific camera at a given distance and the actual distance measured with a meter (reference distance). The standard deviation, on the other hand, was calculated by simply applying the formula to all measurements of the specific coordinate under analysis (in this case, Z).

In some graphs, at certain distances, specific cameras have a red dot at zero. In these cases, it doesn't mean that the standard deviation and absolute error are zero, but rather that the measurement at that distance with that camera was not possible because the marker was not recognized.

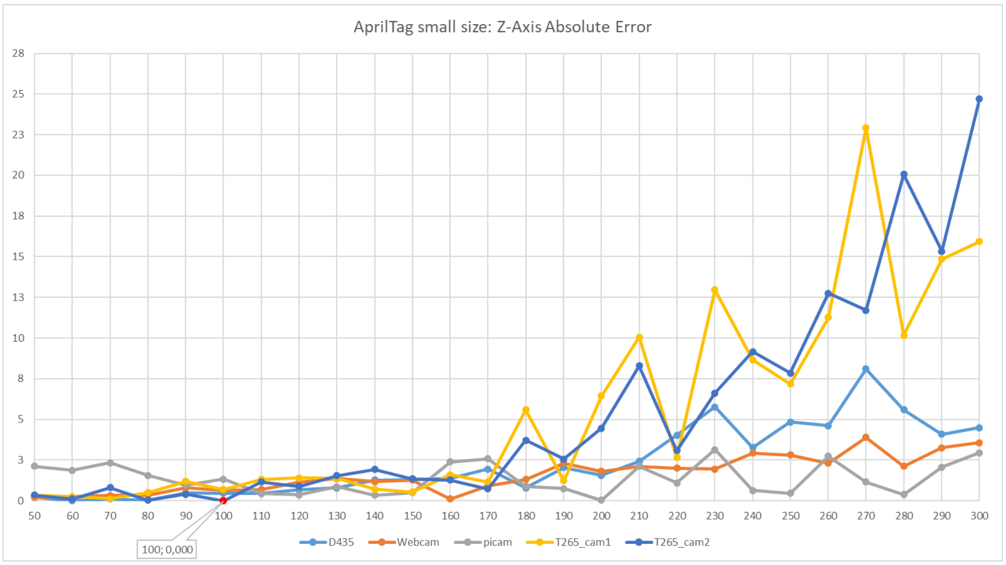


Figure 5.2: AprilTag small size: absolute error of estimated Z distance with respect to the effective one.

AprilTag small size For example, concerning the small AprilTag (fig 5.2 and 5.3), for Lens 2 of the T265, at a distance of 100 cm, the marker was not recognized. However, the peculiar aspect is that this "anomaly" occurred under almost optimal conditions for reading the marker, as the distance of 100 cm is quite close to the

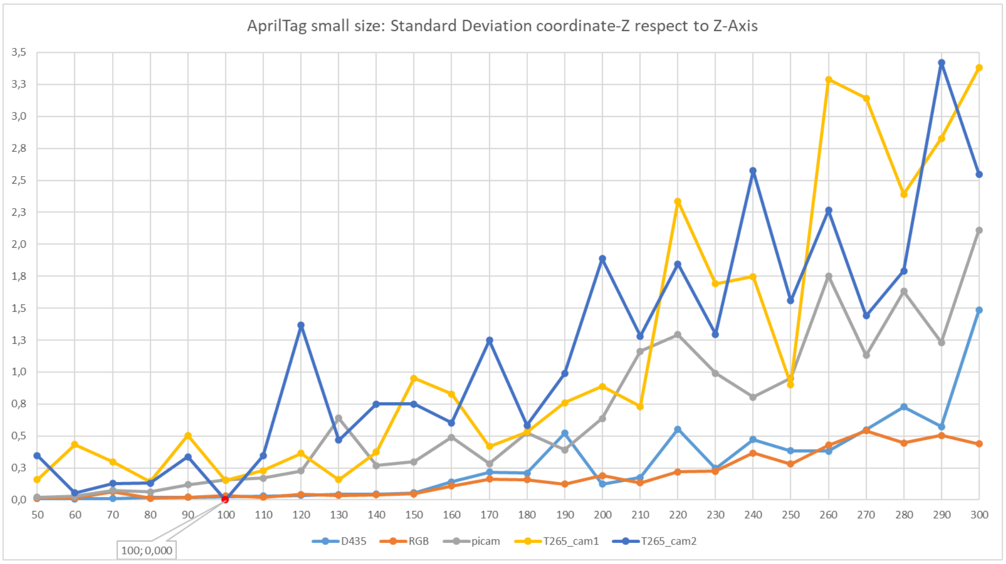


Figure 5.3: AprilTag small size: standard deviation of all the estimated Z coordinates.

marker and significantly below the maximum distance measurable with this camera.

The standard deviation and absolute error for the T265 remain fairly low, constant, and acceptable up to 170 cm; beyond this distance, the measurements indeed start to become imprecise, and both the absolute error and standard deviation assume higher values. The two lenses of the T265, aside from this anomaly at 100 cm, have similar behavior, and the error increases after approximately 170 cm distance.

Regarding the webcam and the D435i, the trend of the absolute error is quite similar: at some distances, one camera has a higher error, while at other distances, the other camera performs worse. After 2 m, the absolute error of the D435i increases slightly compared to that of the webcam Logitech. Concerning the standard deviation, except for some specific distances like 190 cm or 220 cm, the values are quite aligned. In terms of standard deviation, these two cameras yield good results, acceptable compared to the T265 and the Picam, ensuring a certain level of reliability. It is observed that the standard deviation starts to increase around 150 cm but remains well below the average values of the other cameras.

The Picam exhibits a peculiar behavior regarding the small AprilTag. It is, in fact, the camera with the highest absolute error compared to all other cameras, even higher than the T265. The absolute value for the Picam from 70 cm to 120 cm appears to have a decreasing trend, contrary to all other cameras. However, upon observing the standard deviation graph for this camera, we immediately notice that these values are much higher when compared to those of the D435i and the webcam, and slightly lower than the T265. Then, the absolute error seems to vary a bit but is still quite contained, just below the webcam. However, the standard deviation remains very high. This implies that the significantly different values partly compensate each other, yielding an average close to the real distance value, but individual measurements are highly imprecise.

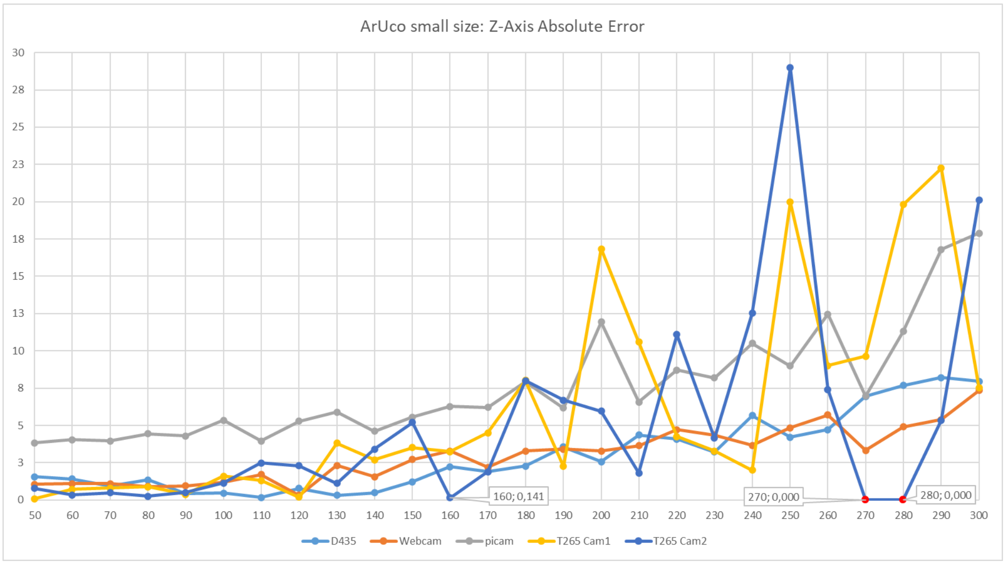


Figure 5.4: Aruco small size: absolute error of estimated Z distance respect to the effective one.

ArUco small size From the graphs 5.4 and 5.5, we observe that, for the small ArUco marker, there are two distances at which Lens 2 of the T265 fails to recognize the marker: specifically at 270 cm and 280 cm. In this case, however, unlike the previous scenario, we are at the limit of the working range of this camera, which extends up to 300 cm. The immediately preceding and subsequent measurements are in fact affected by very high errors, and thus it is expected that the camera may not recognize the marker.

As observed, the performance of this marker is markedly inferior for all cameras. Similar to the previous case, up to 100 cm, almost all cameras show comparable absolute error values. However, from 100 cm onward, the absolute error starts to increase. While for AprilTags, the webcam, and the D435i had absolute errors slightly above 3 cm, in this case, the average absolute error is much higher.

Once again, the Picam stands out for having an absolute error above 3 cm already at 50 cm, demonstrating that it is not very precise and suitable for this use.

However, the key indicator highlighting that the performance and accuracy of

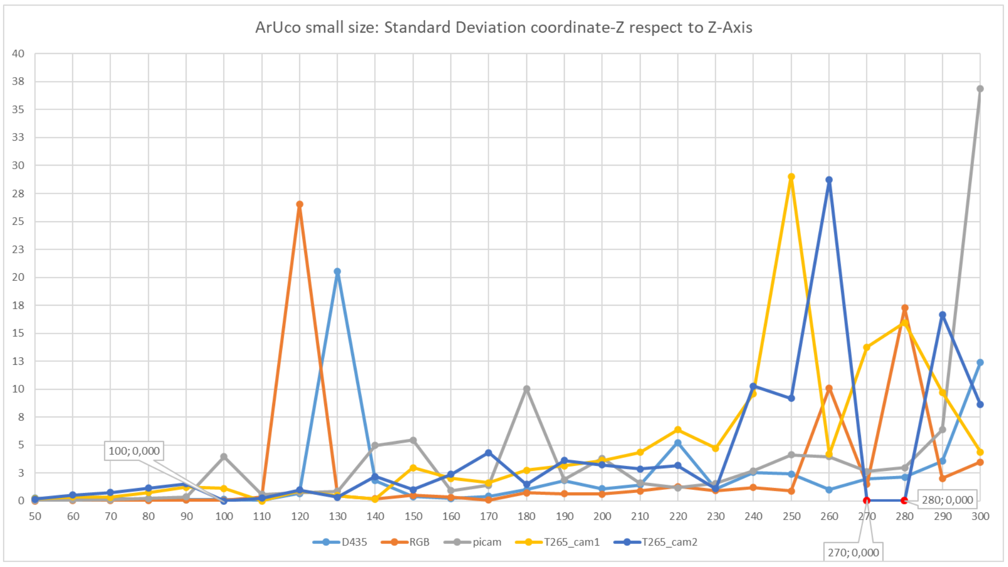


Figure 5.5: Aruco small size: standard deviation of all the estimated Z coordinates.

ArUco compared to AprilTag are much lower is given by the standard deviation. For AprilTags, the standard deviation values were around 0,8 for all cameras, even at approximately 180 cm, reaching a maximum for the T265 of about 3,5 at 290 cm. Conversely, for ArUco markers, the situation is quite different. Standard deviation values are already very high at 150 cm for almost all cameras, reaching peaks above 20 (approaching values of 30) for all cameras (excluding the Picam with a peak of 10).

Considering these peaks and the very high average value, it can be asserted that, for small markers, AprilTags seem to have better accuracy, reliability, and robustness compared to ArUco markers.

AprilTag large size Regarding the larger AprilTags, we have four graphs: two complete graphs 5.6 and 5.7 and two with a limited set of measurements up to 10,5 m 5.8 and 5.9. For this set of measurements, we aimed to push it to the maximum achievable distance to see up to what distance it was possible to recognize the AprilTag (see 5.1.2). Obviously, after a certain distance, the pose estimation became completely unreliable and imprecise, with very high absolute error and standard deviation values that made the measurement graph illegible since the axes were out of scale given the measurement range. For this reason, the graph with measurements limited to 10,5 m is presented to allow a more precise analysis.

Analyzing the complete graph, we observe that precisely at 10,5 m, the PiCam can no longer recognize the marker. However, in any case, the last distance at which it can effectively recognize the marker is 11 m, so we are at the limit of its working

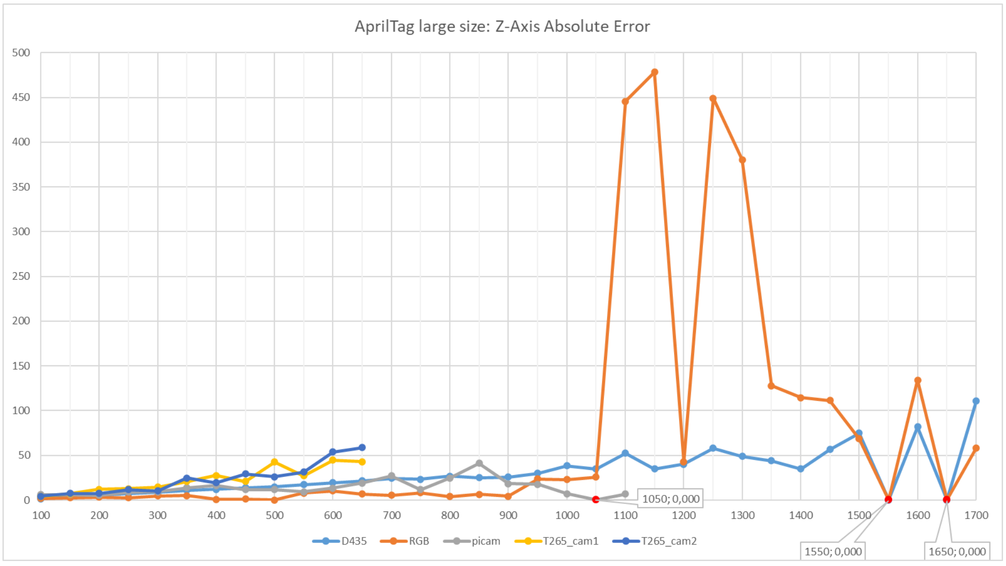


Figure 5.6: AprilTag large size: absolute error of estimated Z distance with respect to the effective one.

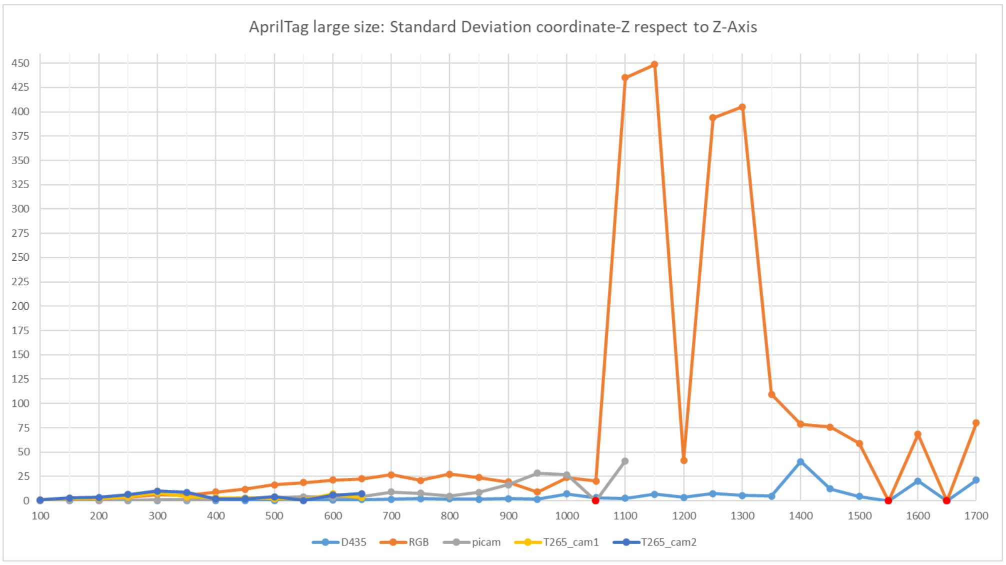


Figure 5.7: AprilTag large size: standard deviation of all the estimated Z coordinates.

range. The webcam and the D435i can recognize the marker up to 17 m, which is a very important distance. Obviously, at these distances, the errors are very high: the absolute error, in fact, hovers around 100 cm, making the measurements completely unreliable. Especially for the webcam, there are four peaks (at 11 m, 11,5 m, 12,5 m, and 13 m) with very high absolute error (around 400-450 cm) and standard deviation values, exceeding the threshold of 400.

Focusing on the two graphs 5.8 and 5.9 with the limited set of measurements, we can analyze and describe the results more accurately.

The camera with the most limited working range and the highest errors is undoubtedly the T265, which already has an absolute error between 10 cm and 15 cm at 300 cm. The maximum working range is around 650 cm with a peak absolute error of 60 cm for lens 2. The standard deviation, all in all, has comparable values to the other cameras, significantly lower than the webcam.

The D435i has a linearly increasing trend for the absolute error, which is not observed in all other cases; there is usually some variation in the trend. Growing consistently and linearly, in this case, makes it easy to notice that as the distance increases, the error grows: at 500 cm, it's around 15 cm, while at 10,5 m, it's around 35 cm, which is quite significant. On the other hand, the standard deviation is quite constant, increasing slowly. This means that the measurements, beyond an error relative to the reference distance, are quite accurate and close to each other. At 9,50 m, the standard deviation is about 2,5, while at 550 cm, it is 0,33. Therefore, the pose measurements of the D435i are quite good.

The webcam has a particular behavior: by examining the graph of absolute error, fluctuating values are evident, yet they never exceed the absolute error of the D435i. Even beyond 5 m, most measurements have an absolute error below 10 cm, peaking at approximately 20 cm at 10 m distance. However, when observing the graph illustrating the standard deviation, rapid growth in values is observed, surpassing 25. This indicates significant differences among measurements. While compensating for each other the absolute error is limited, but individual measurements are consequently not very precise.

Finally, the PiCam has absolute error values that oscillate quite a bit, and after 6 m, they start to grow significantly. If we associate this with the standard deviation, which also starts to grow from 6,5 m, we can deduce that, in this case again, the measurements are not very precise for this camera.

ArUco large size Finally, observing the graphs of the large ArUco marker 5.10 and 5.11, it is immediately noticeable that the performance is more discontinuous compared to the AprilTag.

For this marker as well, the two lenses of the T265 have a limited working range compared to the other cameras: lens 1 can recognize the marker at a distance of 6 m, while lens 2 reaches 5,5 m. It is interesting to note that both lenses fail to recognize the marker at the "penultimate" distance of the working range, then recognize it immediately afterward with a very high absolute error: indeed, both lenses exceed 60 cm. The standard deviation for this camera is overall similar to other cameras, except for a peak of about 30 for lens 2 at 200 cm, but apart from

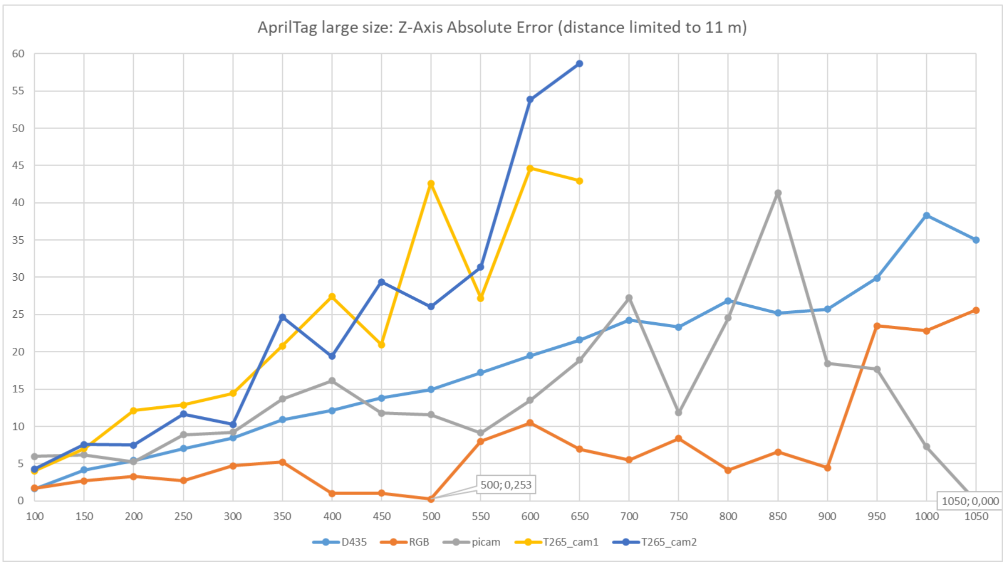


Figure 5.8: AprilTag large size: absolute error of estimated Z distance with respect to the effective one. The number of measures is limited to have a better plot axis range.

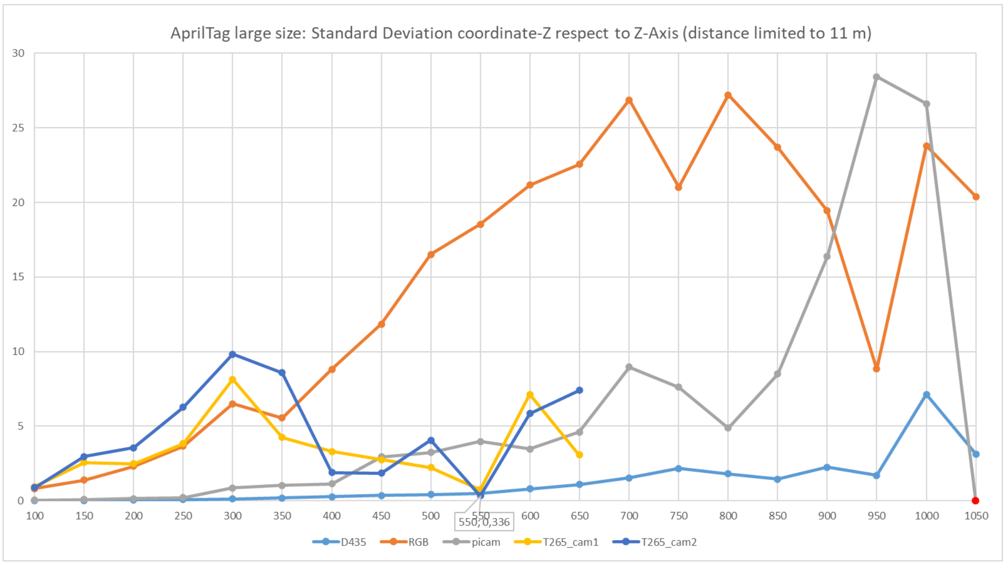


Figure 5.9: AprilTag large size: standard deviation of all the estimated Z coordinate. The number of measures is limited to have a better plot axis range.

this, there are no particularly high values.

The Picam, as in all other measurement sets, does not excel particularly. Up to 350 cm, it has an absolute error of around 5 cm, in line with other cameras. Then, as the distance increases, the absolute error grows significantly, and the standard deviation also increases in the same way. The two quantities grow consistently with each other.

Regarding the webcam and the D435i, these two cameras have similar behavior in terms of absolute error. There is no linear trend as with the AprilTags, but the absolute error fluctuates significantly between values around 5-10 cm, with those of the D435i slightly higher. In this case, there was no need to select a limited set of distances, as beyond 12 m, the cameras were no longer able to recognize the ArUco marker, unlike the AprilTag, which was recognizable even at 17 m. The webcam, even at distances of 11 m, has relatively acceptable absolute error values, around 2,5 cm, and standard deviation around 15, so not excellent results but the distance is still important.

The standard deviation of the D435i suddenly increases at 6 m, reaching 95, then gradually decreases to values very close to 0 at 8,5 m. This is a particularly curious phenomenon: in this measurement, marker recognition was perfectly stable, with no micro variations recorded. It is a unique condition that occurs at certain specific distances, even 8,5 m and 11,5 m, and the reason is not explained. The standard deviation of the webcam, on the other hand, starts to grow gradually from 7,5 m.

In this case, as well, the two most promising cameras seem to be the D435i and the webcam.

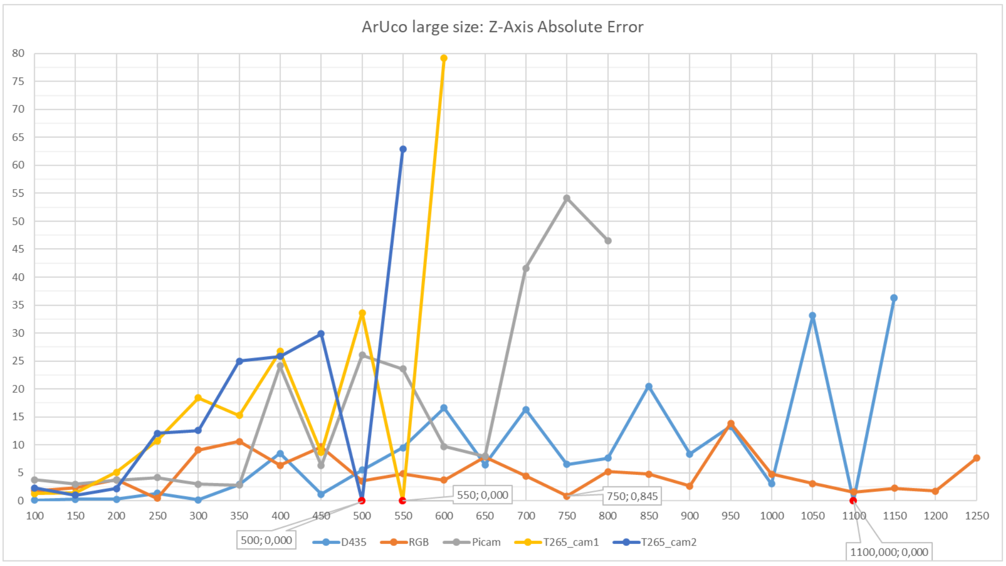


Figure 5.10: Aruco large size: absolute error of estimated Z distance with respect to the effective one.

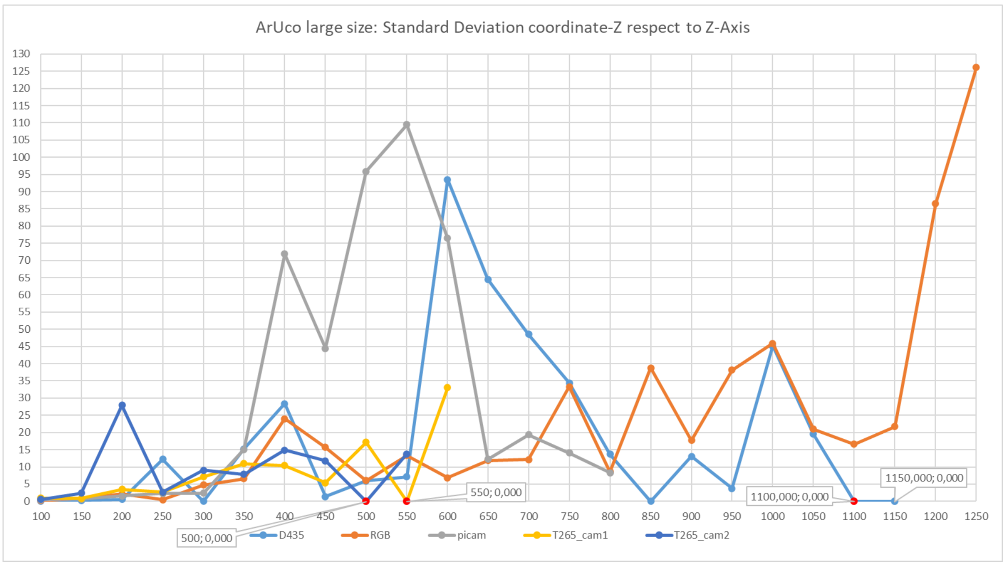


Figure 5.11: Aruco large size: standard deviation of all the estimated Z coordinates.

General consideration on Z-Axis analysis Taking an overall consideration of different cameras and markers, we can assert that in terms of performance, AprilTag seems to be significantly superior to ArUco. Its working range is larger, and the absolute error and standard deviation are generally lower, with better recognition stability for AprilTag.

Among the cameras, those that stood out for better performance in terms of distance are the webcam and D435i, which generally recorded larger working ranges and greater accuracy.

As observed, many cameras have reasonably good (and still acceptable) values up to 150 cm - 170 cm. For this reason, the distance of 160 cm was chosen for the analysis on the X and Y axes for small markers, and also because the idea is to simulate navigation and relocation outside the corridors. The distance of 70 cm, on the other hand, considers a scenario where the drone is at the center of the corridor at a distance of about 70 cm from the shelf.

Regarding the large markers, the distance of 450 cm was chosen because generally, below 5 m, all cameras were able to recognize both markers, representing a good compromise between distance and accuracy.

Example of theoretical max distance computation From [31] the theoretical max distance to detect an Apriltag is computed as:

$$Max_{distance}[m] = \frac{t}{2 * tan(\frac{b*f*p}{2*r})}$$

where

- t is the size of the tag (in meters)
- b is the number of bits that span the width of the tag (excluding the white border for Apriltag 2). In our case for AprilTag 36h11 = 8
- f is the horizontal FOV of the camera
- r is the horizontal resolution of the camera
- p is the number of pixels required to detect a bit. This is an adjustable constant. It is recommended 5 (to avoid some of the detection pitfalls). The lowest suggested number is 2 that is the Nyquist Frequency.

So considering the cameras:

- Intel RealSense - D435i
  - RGB max frame resolution: 1920 × 1080 (it is possible, eventually to use reduced resolution set)
  - RGB sensor FOV (H × V x D): 69° x 42° x 77°
  - RGB sensor resolution: 2 MP
- Logitech webcam HD C270
  - Sensor FOV 55°
  - Frame resolution 1280 x 720
  - Resolution 0.9 Megapixels

Computing the max theoretical distance for D435i we obtain:

- with $p = 5 \rightarrow$ 11,15m
- with $p = 4 \rightarrow$ 13,94m
- with $p = 3 \rightarrow$ 18,60m
- with $p = 2 \rightarrow$ 27,90m

and for webcam:

- with $p = 5 \rightarrow$ 9,33m
- with $p = 4 \rightarrow$ 11,66m
- with $p = 3 \rightarrow$ 15,55m

- with $p = 2 \rightarrow 23{,}33m$

Considering the obtained result, the theoretical max distance is reasonable. In the library that recognizes the AprilTag is not possible to define the p value, but estimating it around 3 the obtained max distance is totally coherent and reasonable.

Pose estimation accuracy: X-Axis

Now let's analyze the cameras' behavior along the X-axis. It's worth noting that only the most significant graphs have been chosen for presentation effectiveness, but measurements have been taken for both markers and all cameras at 70 cm and 160 cm for small markers, and 450 cm for large ones.

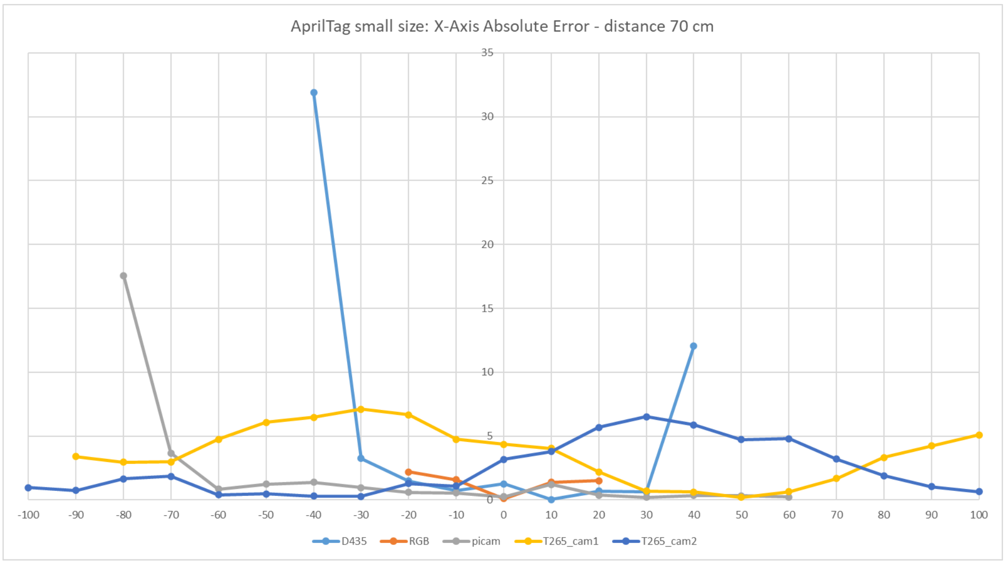


Figure 5.12: AprilTag small size: absolute error of estimated X at a distance of 70 cm from the marker.

Before starting the analysis, it is important to note that the camera working ranges are not exactly symmetrical due to restrictions in the laboratory area. The limited space prevented symmetric measurements. This asymmetry is evident in the X and Y analyses since we installed the two markers side by side. However, it did not impact the Z-axis analyses.

The measurements, though asymmetrical, were sufficient for understanding how these cameras and markers behave and drawing general conclusions.

Looking at the graphs of small markers at close distances of 70 cm, the first thing observed is the working range of different cameras. As expected, the range width is opposite to the measurements made along the Z-axis. The webcam and D435i, with narrower fields of view compared to T265 and wide-angle Picam, have

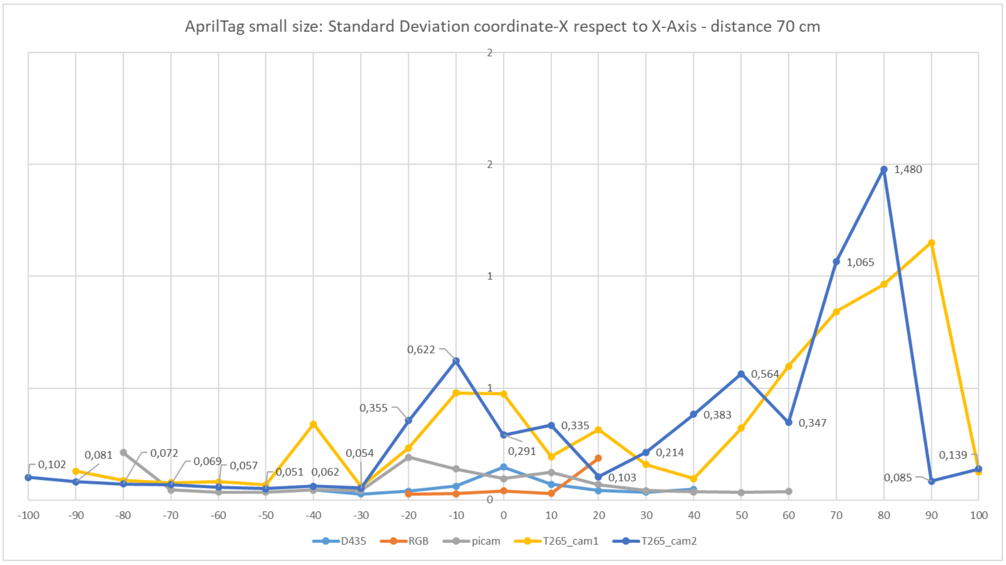


Figure 5.13: AprilTag small size: standard deviation of all the estimated X coordinates at a distance of 70 cm from the marker.

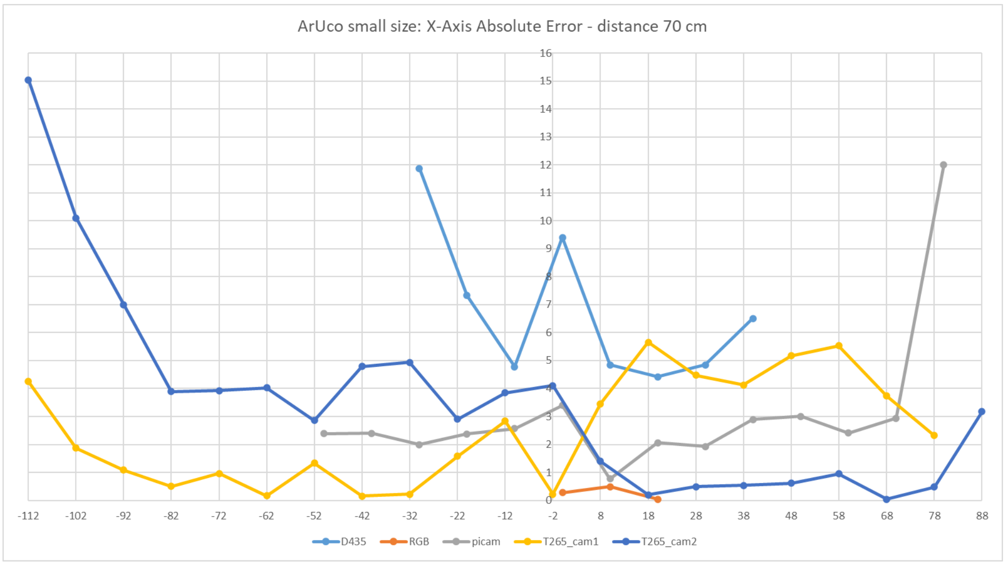


Figure 5.14: Aruco small size: absolute error of estimated X at a distance of 70 cm from the marker.

a limited working range. The webcam, for example, has a range from -20 cm to +20 cm for AprilTag and only 20 cm for ArUco. The D435i has a slightly wider range (-30 cm to +40 cm), while the Picam spans from -80 cm to +60 cm. The T265, with fisheye optics, operates in a range from -100 cm to +100 cm for AprilTag and

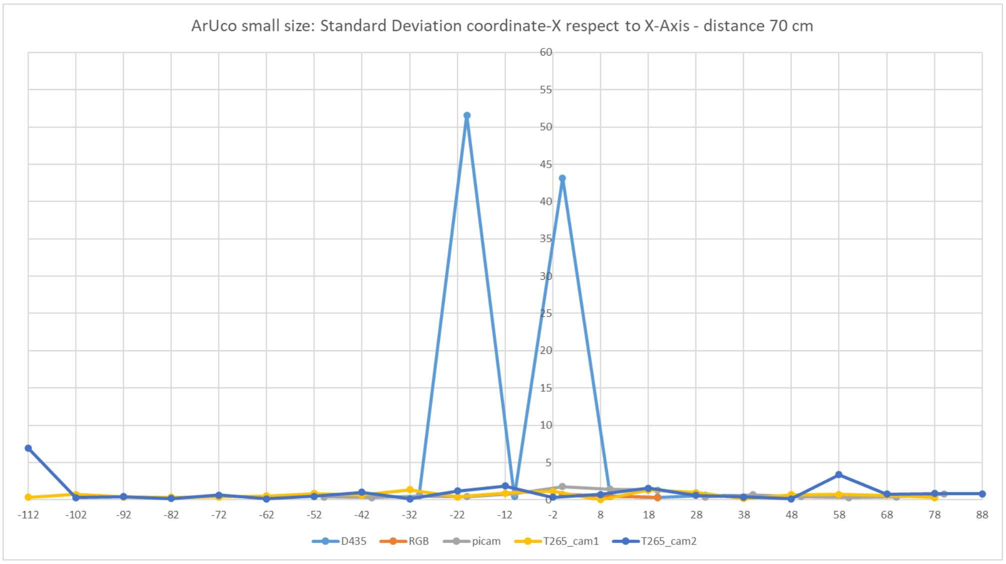


Figure 5.15: Aruco small size: standard deviation of all the estimated X coordinates at a distance of 70 cm from the marker.

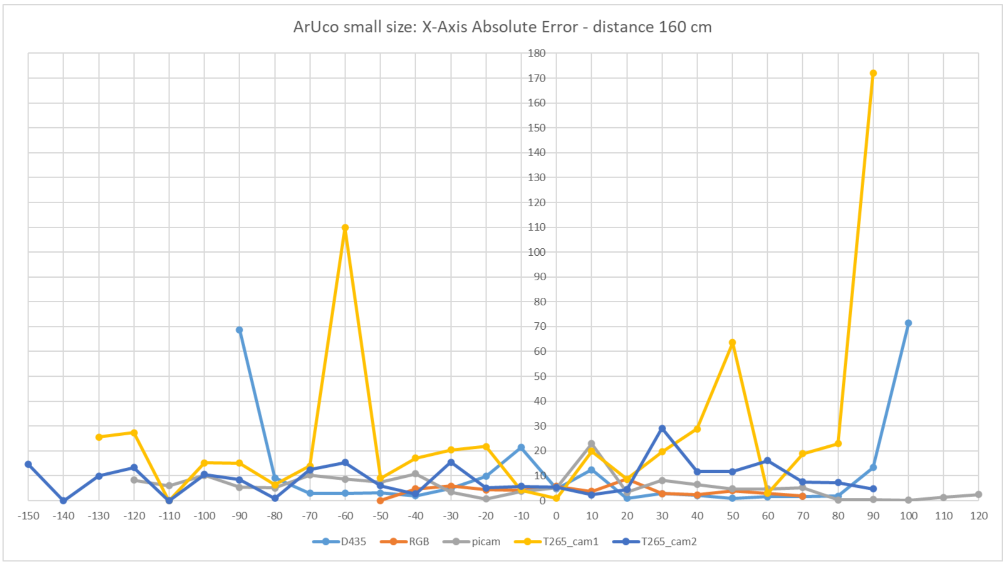


Figure 5.16: Aruco small size: absolute error of estimated X at a distance of 160 cm from the marker.

from -110 cm to +90 cm for ArUco.

As mentioned at the beginning of the chapter, these asymmetries are not due to camera anomalies but rather result from physical constraints in the laboratory area. The limited space prevented free movements of the test bench without constraints.

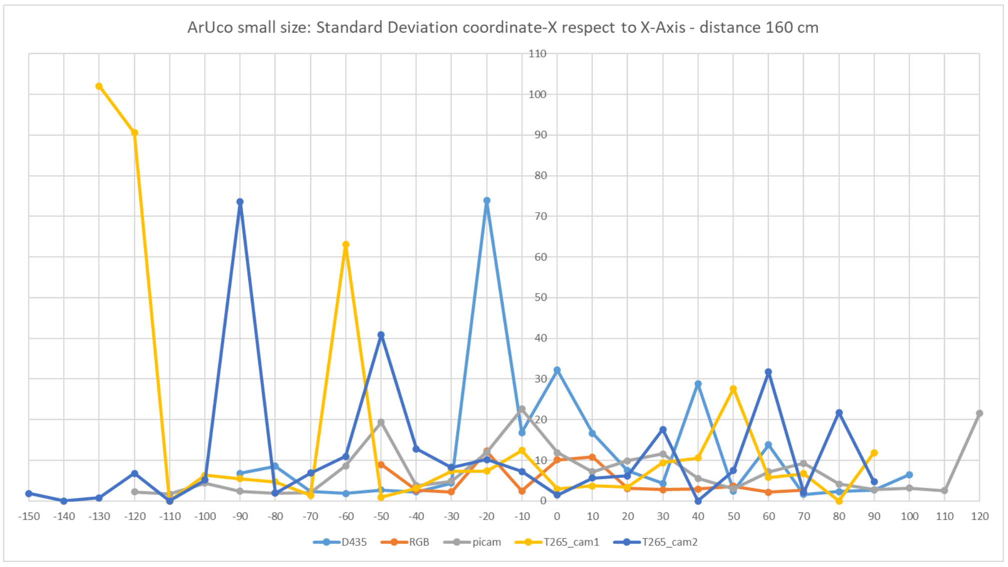


Figure 5.17: Aruco small size: standard deviation of all the estimated X coordinates at a distance of 160 cm from the marker.

However, our analyses are not influenced or impacted by this asymmetry.

Interestingly, contrary to expectations, many cameras tend to have high absolute error and standard deviation values around zero (perpendicular to the marker). These values then decrease at intermediate distances within the working range, only to rise again at the edges of the working range.

This phenomenon is particularly noticeable for the D435i and Picam in the ArUco graphs at a distance of 70 cm (Figures 5.14 and 5.15) and 160 cm (Figures 5.16 and 5.17). With AprilTag at 70 cm (Figures 5.12 and 5.13), a similar trend is observed for the two lenses of T265 and D435i.

Focusing on the absolute error graphs for AprilTag (Figure 5.12) and ArUco (Figure 5.14) at 70 cm, a peculiar behavior regarding the absolute error of the T265 lenses is noticeable. There is an antisymmetry with respect to the origin. In 5.12 lens 1 exhibits greater accuracy for positive values ($X > 0$) and greater error for negative values ($X < 0$) while lens 2 has exactly the opposite behavior: greater accuracy for negative values ($X < 0$) and greater error for positive values ($X > 0$). The maximum error occurs at approximately -30 cm (Lens 1) and +30 cm (Lens 2). At around 10 cm (close to 0), both lenses have nearly identical values, with Lens 1 = 4,025 and Lens 2 = 3,797. The error then rises again around -60 cm (Lens 2) and +60 cm (Lens 1). As of now, there is no explanation for this phenomenon. In the case of ArUco, the phenomenon is less precise and evident, but both lenses exhibit opposite behavior: Lens 1 is more accurate for $X < 0$, and Lens 2 seems more precise for $X > 0$.

In general, absolute error and standard deviation values are reasonably contained and acceptable. For AprilTag, at a distance of 70 cm, the average error for all cameras is less than 5 cm, with higher peaks only at the edges of the working range. The standard deviation for AprilTag (Figure 5.13) has the lowest values recorded among all measurements, indicating that the measurements are very precise, stable, and similar to each other. The maximum value is even less than 1,6, while the average is less than 0,5.

For ArUco, the absolute error is higher, confirming the superior performance of AprilTag. In Figure 5.15, two peaks of very high standard deviation are noticeable for the D435i, exceeding 40. This once again confirms that measurements around the perpendicular axis are less precise, likely due to Z-flipping. However, apart from these two peaks, the standard deviation has very low values, less than 2,5, confirming good performance for ArUco at 70 cm too.

As expected, performance significantly decreases for measurements at a distance of 160 cm for both markers. However, the general behaviors observed at 70 cm are still present at 160 cm.

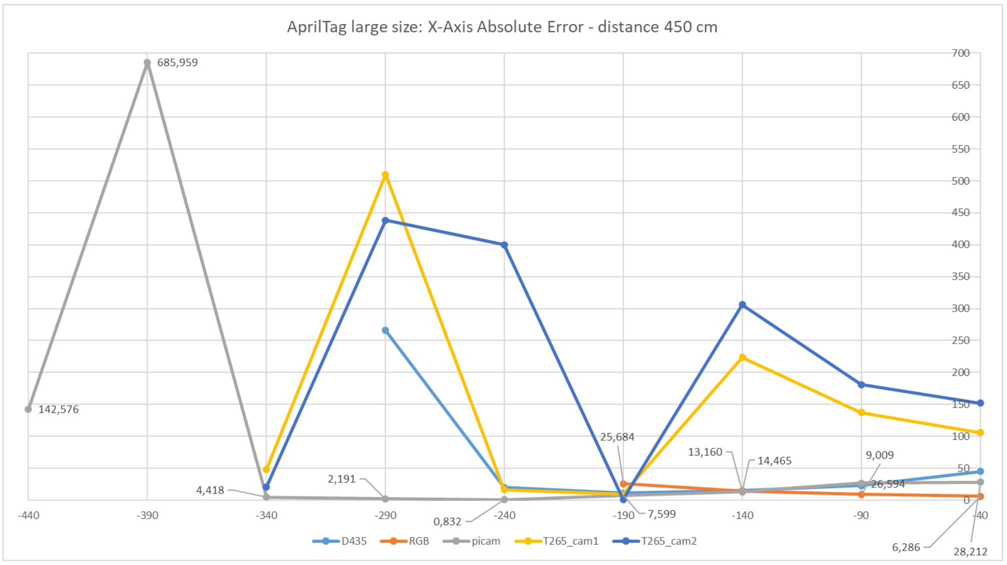


Figure 5.18: AprilTag large size: absolute error of estimated X at a distance of 450 cm from the marker.

Analyzing the measurements for the larger markers at a distance of 450 cm, it is expected that increasing the distance will expand the range of work, especially for cameras with a wider FOV such as Picam and T265. Observing data from measurements of the smaller markers, which exhibited a certain symmetry (and in some cases, an unclear antisymmetry) of measurements with $X > 0$ and $X < 0$, considering the spatial limitations of the measurement area, it was decided to move

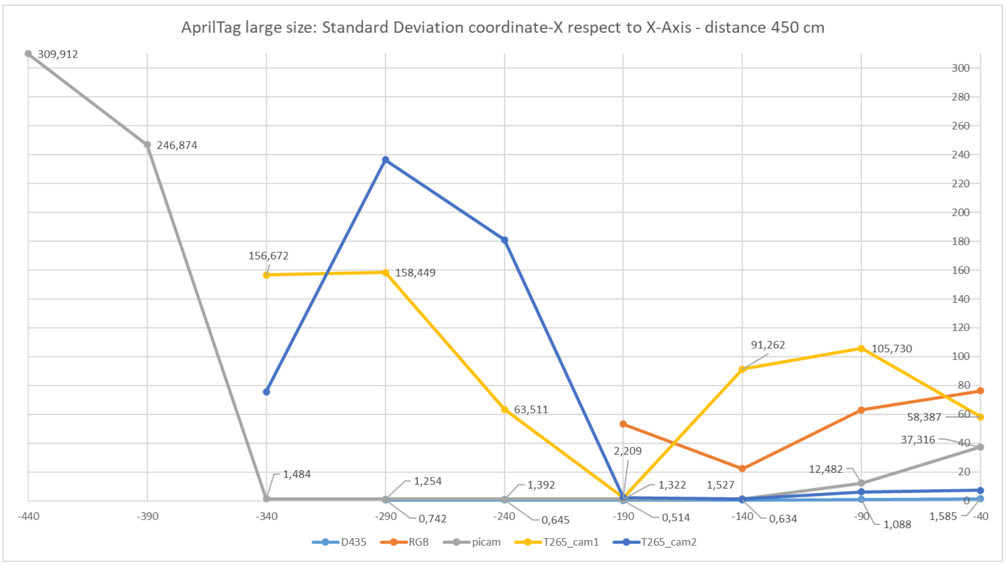


Figure 5.19: AprilTag large size: standard deviation of all the estimated X coordinates at a distance of 450 cm from the marker.

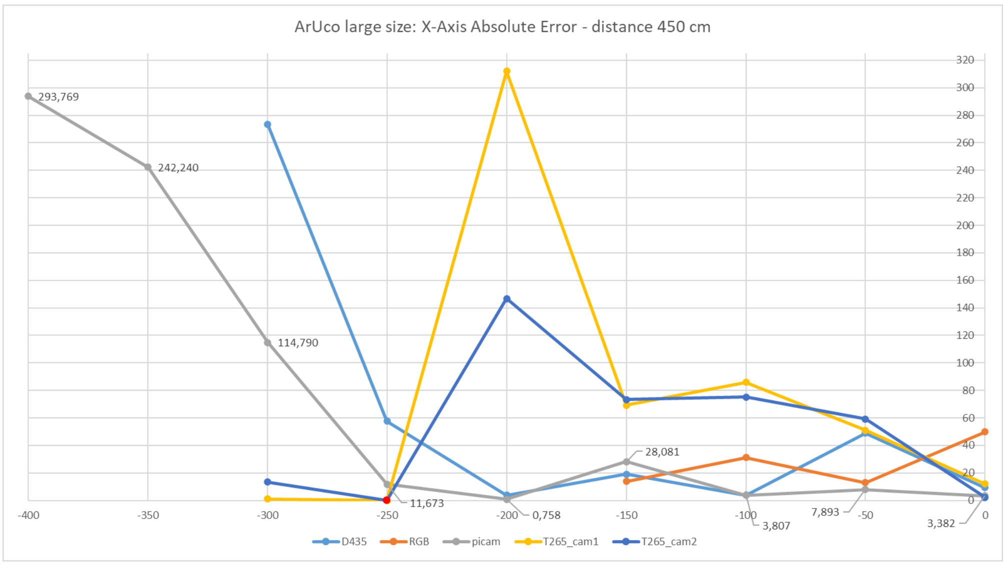


Figure 5.20: Aruco large size: absolute error of estimated X at a distance of 450 cm from the marker.

the markers to one side of the area. Measurements were then taken only in one direction, optimizing a single direction and reaching the maximum working range, even with wide-angle and fisheye lenses.

Firstly, let's observe the working range. Distances for measurements are not

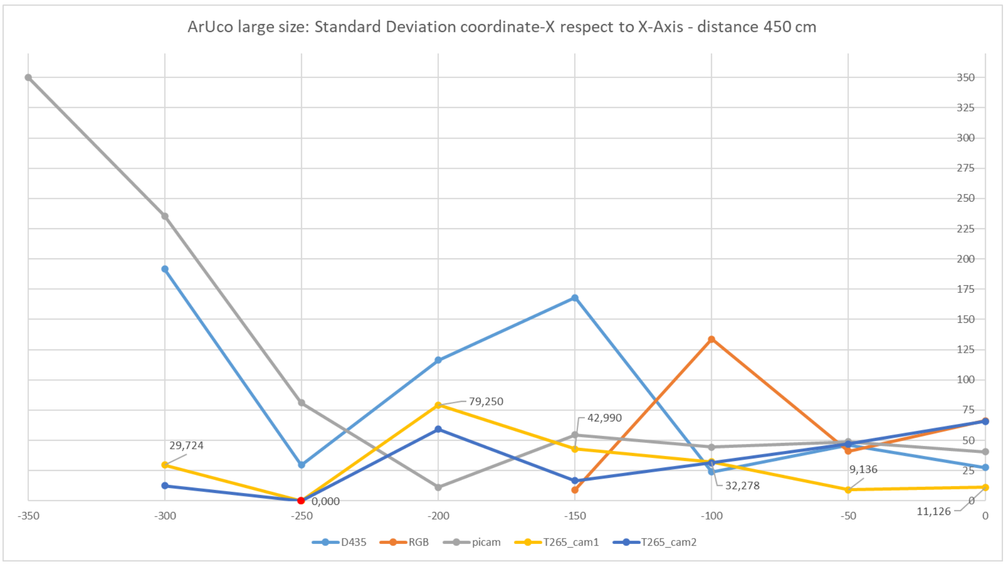


Figure 5.21: Aruco large size: standard deviation of all the estimated X coordinates at a distance of 450 cm from the marker.

exactly the same, this is due to space limitations. Having the two markers positioned side by side, there is an initial translation of 40 cm for one marker, while the other is centered in 0. Despite this, the working ranges of the cameras are almost the same, with the only difference being that, due to this initial translation, the D435i has one less measurement for AprilTags, reaching -290 cm, while for ArUco, starting from 0 cm, it reaches -300 cm. This is because, in the next measurement (at -340 cm for AprilTags and -350 cm for ArUco), the marker went out of the camera's FOV.

Despite these small differences in the measurement ranges, it is observed that, unlike the previous case where the only limit to the working range was determined solely by the FOV of the cameras for smaller markers at close distances, favoring the Picam and especially the T265 with fisheye optics, in this case, with an increased distance between the camera and the marker, resolution also plays a role in the working range. Cameras like the webcam and D435i, as we move away from the marker, also increase their working range. In fact, the D435i has almost the same range as the T235 (which has "only" 50 cm more range), whereas in the previous case, there was much more difference between the D435i and the T265 in proportion. In this case, the camera with the maximum range is no longer the T265 but the Picam because it combines a wide-angle lens with a high FOV with a resolution and image quality superior to the T265. Due to the distortion of the lens, the T265 significantly impacts image quality at greater distances, affecting the maximum horizontal deviation at which the camera recognizes the markers.

The Picam, therefore, has a wider working range of up to 4,4 m for AprilTags and 4 m for ArUco (the difference is always due to the position of the markers). However, at the limits of the range, the absolute error is very high, reaching a peak of 680 cm for AprilTags. Before 3,4 m, for AprilTags, very constant and low absolute error values are recorded, around 2 cm 5.18. Regarding ArUco, the range of acceptable values is reduced to 2,5 m. In this case, the absolute error has more irregular values, around 10 cm on average 5.20. Also, for the standard deviation, the AprilTag 5.19 has, on average, slightly lower values than ArUco 5.21. In both cases, once the two thresholds of 3,4 m (for AprilTag) and 2,5 m (for ArUco) are exceeded, the standard deviation also increases significantly, concurrently with the absolute error.

The T265 with the AprilTag marker exhibits a very high absolute error, already exceeding 100 cm at -40 cm, as shown in 5.18, for both optics. The corresponding standard deviation 5.19 mirrors the values of the absolute error, especially for optic 1, while optic 2 has a relatively contained standard deviation. There is a noticeable decrease in both absolute error and standard deviation at -190 cm. Regarding ArUco, the situation is similar; absolute error and standard deviation have low values only for the first measurement at 0 cm offset from the Z-axis, and then the graphs steeply rise as we move laterally 5.20 5.21.

The webcam and D435i also show generally lower and more regular absolute error and standard deviation measurements with AprilTags. It is worth noting that the D435i with AprilTags has a very regular and low absolute error up to -240 cm, increasing only for the last measurement within its working range. The standard deviation remains consistently low for the entire range of measurements, with values always below 2.

In this scenario, the cameras that demonstrated the best performance in terms of accuracy, stability, and working range were the Picam with the largest range and good precision, followed closely by the D435i, with a slightly smaller range (with less difference for range measurements at 450 cm distance) and very precise measurements.

Pose estimation accuracy: Y-Axis

For the analysis of measurements on the Y-axis, as in the previous case, the most representative measurement sets were chosen to provide a comprehensive view of the behavior of various cameras and markers in this comparison. However, for the analysis, all measurement data at distances of 70 cm and 160 cm for small markers and 450 cm for large markers were considered.

In this final scenario of the comparison, it is expected that by rotating the cameras by 90° to conduct measurements laterally, on the horizontal plane parallel to the floor, the resolution and FOV of the cameras are reduced. Consequently, the working range and accuracy are expected to be lower than those along the X-axis.

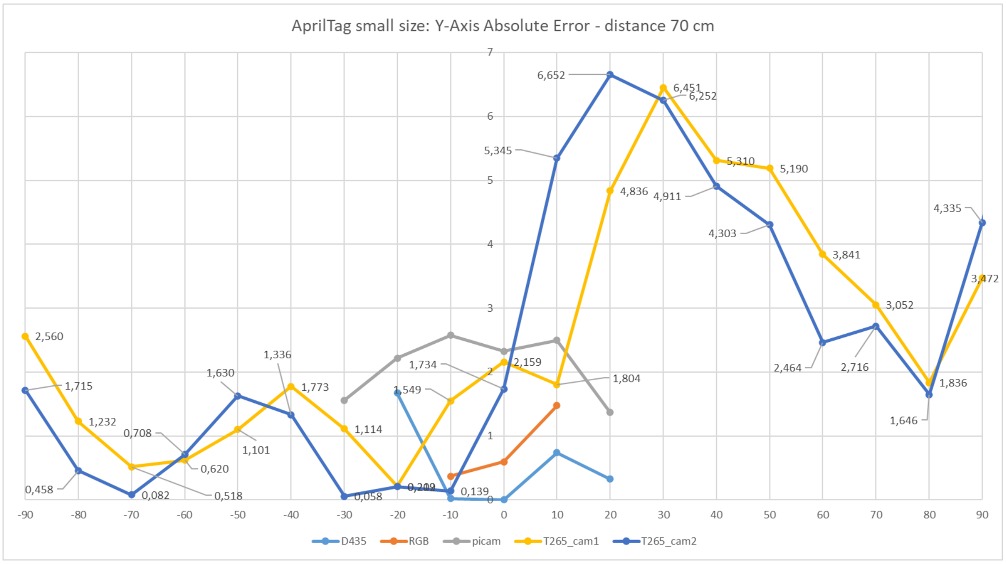


Figure 5.22: AprilTag small size: absolute error of estimated Y at a distance of 70 cm from the marker.

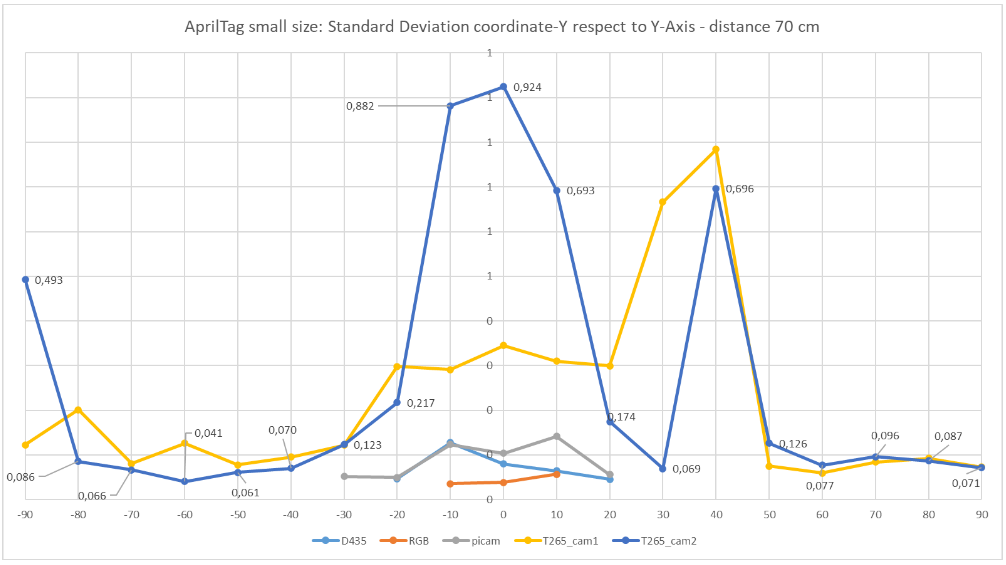


Figure 5.23: AprilTag small size: standard deviation of all the estimated Y coordinates at a distance of 70 cm from the marker.

From the graph 5.22, a peculiar antisymmetry can be observed in the behavior of the absolute error of lenses 1 and 2 of the T265 between $Y > 0$ and $Y < 0$.

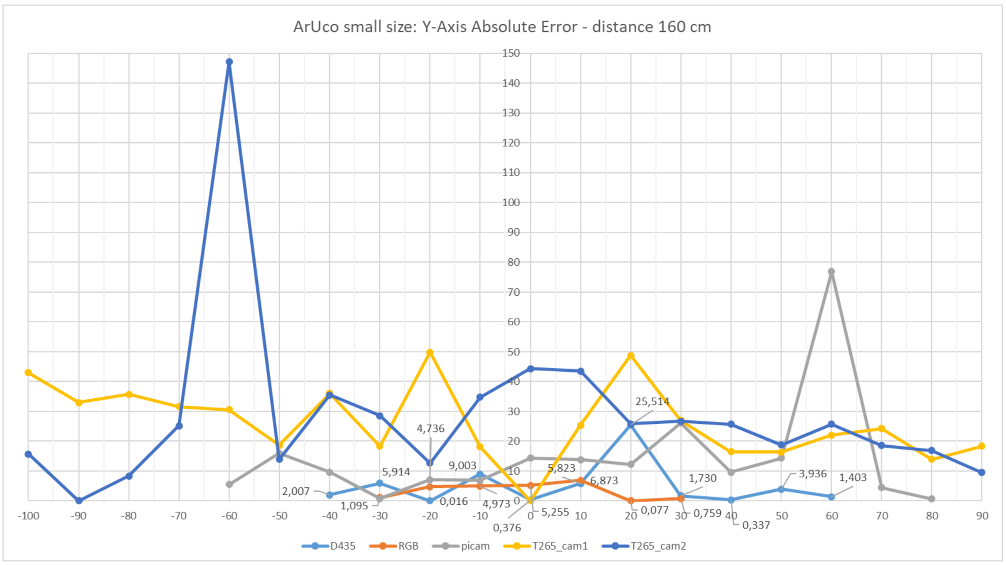


Figure 5.24: Aruco small size: absolute error of estimated Y at a distance of 160 cm from the marker.

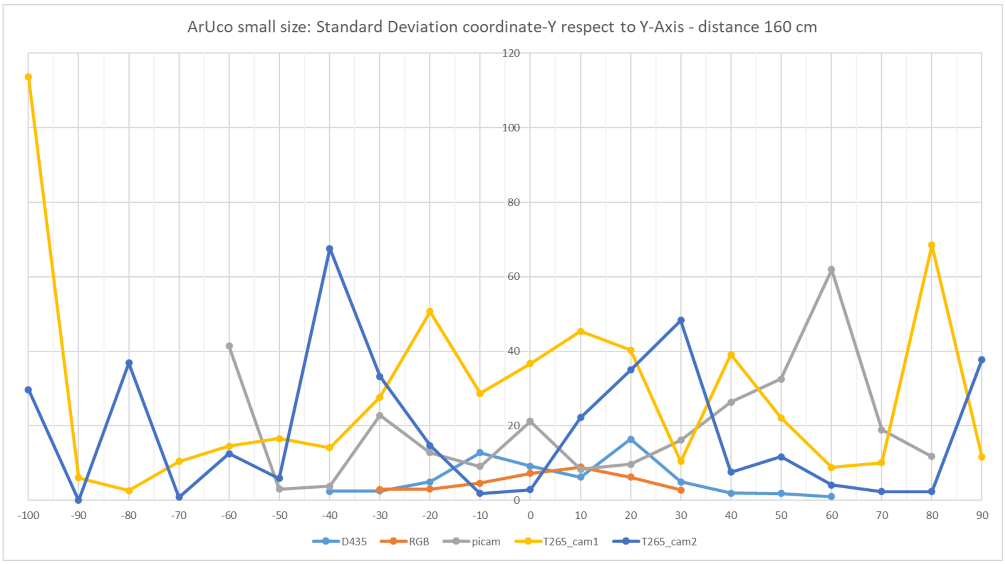


Figure 5.25: Aruco small size: standard deviation of all the estimated Y coordinates at a distance of 160 cm from the marker.

Unlike the X-axis, in this case, the absolute error behaves in the same way for both lenses (due to the 90° rotation of the camera), and it is interesting to note that the absolute error of the two lenses, although they are at the same Y, seems to be offset by 1-2 cm.

Regarding the standard deviation, it is interesting to note from 5.23 the same phenomenon found for the X-axis, i.e., the error and standard deviation increase at the perpendicular, i.e., the closer one is to 0. This behavior is also found in the two ArUco graphs at a distance of 160 cm 5.24 and 5.25.

In general, the performances are confirmed to be better for all cameras at both distances of 70 cm and 160 cm in favor of AprilTags, which ensures more accurate and stable measurements.

It is noteworthy that all cameras, especially at 70 cm, have a very limited working range, given the lower FOV and resolution along the Y-axis (vertical). However, for the T265, which has a fisheye lens and a very wide FOV, the working range remains practically unchanged, being from -90 cm to +90 cm. The working range of the Picam, despite having a wide-angle lens, is very close to that of the webcam and D435i, it is around -30 cm to +20 cm.

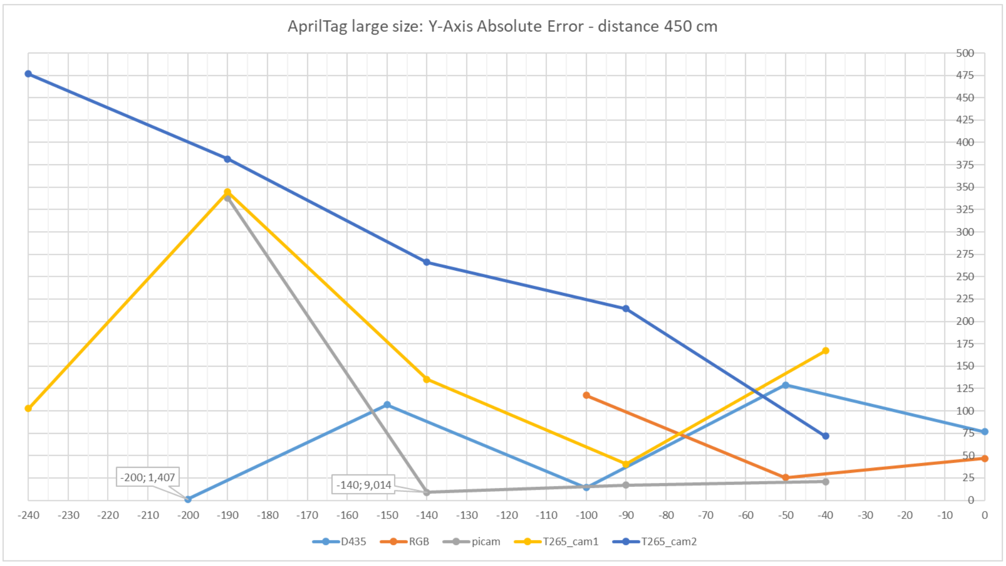


Figure 5.26: AprilTag large size: absolute error of estimated Y at a distance of 450 cm from the marker.

Regarding the measurements with the large markers at a distance of 450 cm, there is not much to say: the measurements are very noisy, the error is very high, and, as with the measurements on the X-axis, they are generally unreliable and imprecise. Therefore, it is probably not suitable for relocation.

Furthermore, as can be seen from 5.28 and 5.29, for lens 1 of the T265, the camera apparently cannot recognize the marker in any position of this set of measurements. This is an unexpected result because the same lens on the X-axis at the same distance was able to recognize the ArUco, and lens 2 was able to recognize it. However, this certainly supports the thesis that marker identification with this

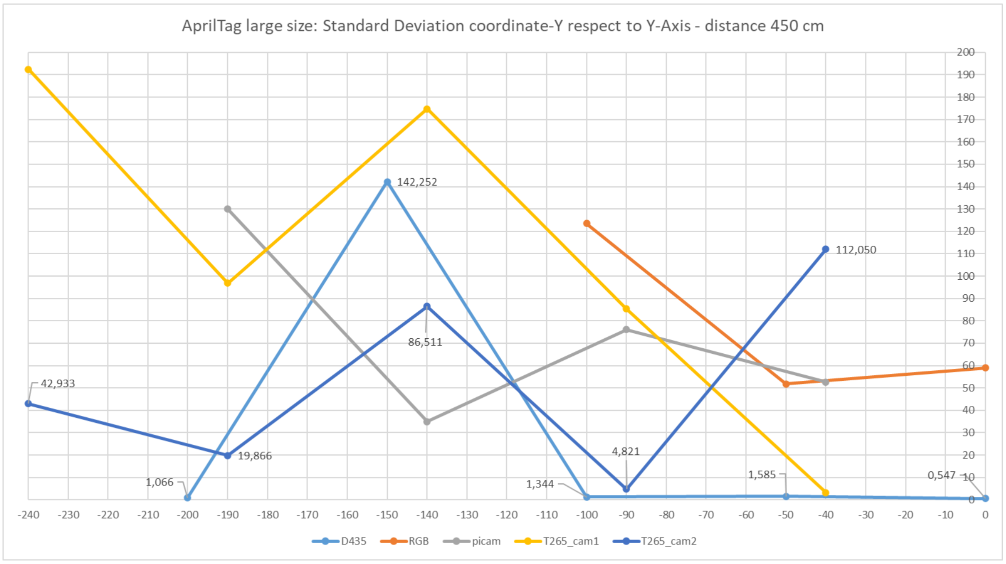


Figure 5.27: AprilTag large size: standard deviation of all the estimated Y coordinates at a distance of 450 cm from the marker.

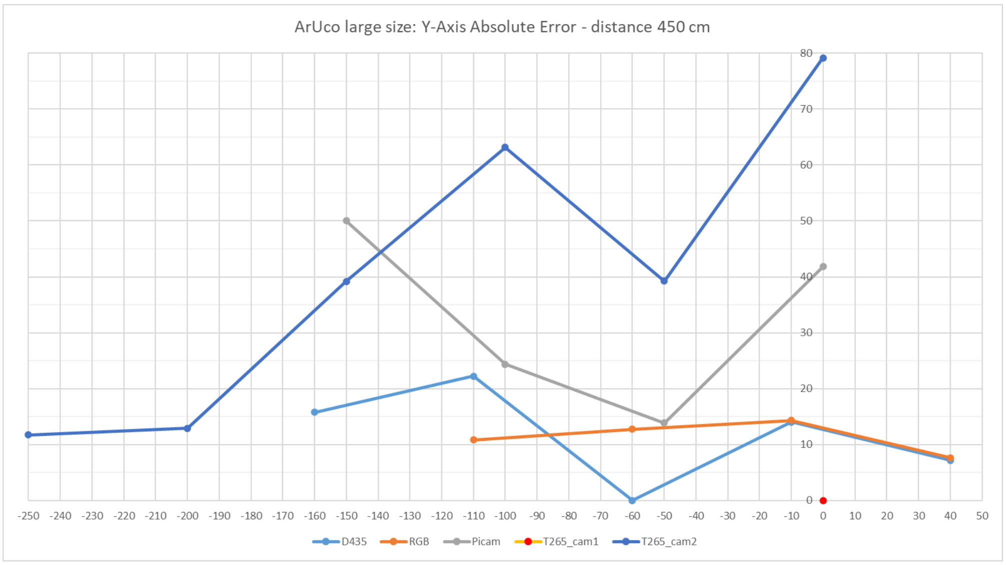


Figure 5.28: Aruco large size: absolute error of estimated Y at a distance of 450 cm from the marker.

camera at this distance is quite unreliable.

As with the small markers, in this case, as well, the working range of the Picam has been drastically reduced, being more similar to the one of the webcam and D435i. However, even in this case, the measurements are still very imprecise.

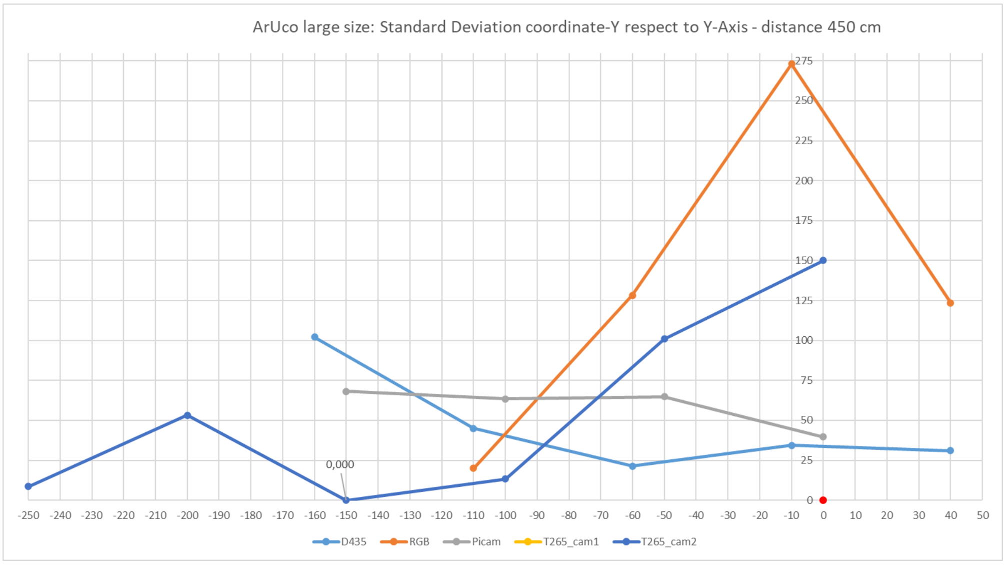


Figure 5.29: Aruco large size: standard deviation of all the estimated Y coordinates at a distance of 450 cm from the marker.

5.2 Conclusion & future works

Having analyzed all the measurement sets for the X, Y, and Z axes, conclusions can now be drawn from this comparison.

In general, AprilTags demonstrates better performance across all cameras. Specifically, they have a broader working range on the Z-axis, facilitating marker identification even at greater distances. Moreover, stability and accuracy generally surpass those of ArUco markers. As noted initially, AprilTags also exhibit a higher pose stream frequency under similar conditions, indicative of robustness and reliability.

Considering the results for individual cameras, it can be concluded that the camera that appears most suitable for reliable and precise relocalization, while consistently maintaining a high data stream frequency, is the D435i. This camera, distinguished by its balanced performance across various metrics, boasts the highest frequency alongside the T265. Within its working range, measurements prove reliable and precise. Striking a balance between the webcam with a very limited field of view (FOV) and the T265 or Picam with an unnecessarily (for this specific use case) wide FOV, the D435i's extensive working range sets it apart. While cameras with wide-angle lenses offer a wide FOV, they are susceptible to significant distortion, impacting marker identification accuracy even if it is placed in the camera's FOV. A limited FOV, such as that of the webcam, may have troubles when is very close to the marker plane (as at 70 cm distance) because limited visibility affects marker detection likelihood.

In our scenario, this camera is likely one of the most suitable options. To enable a drone to navigate autonomously within an indoor environment, various techniques, as discussed in Chapter 3, can be employed. However, these techniques require sensors that provide necessary data to Kallman filters and algorithms for pose estimation, trajectory calculation, and obstacle detection. In this application field, a comprehensive and robust solution should not only be capable of self-localization and navigation but also recognize potential obstacles.

Onboard a drone, the use of Lidar or similar systems is discouraged due to vibrations that would affect data quality and pose challenges in terms of weight and energy consumption. Hence, a camera like the D435i, lightweight, compact, energy-efficient, and equipped with a CPU, becomes ideal for this use case. With a single sensor, obstacle avoidance can be enabled through the Depth module (which gives the "D" to the Intel RealSense camera family). Additionally, using the RGB module (utilized in this experiment), relocalization with markers becomes possible.

In conclusion, with AprilTags and the D435i camera, it can be considered to implement relocalization based on fiducial markers, provided the algorithm is structured appropriately. As observed for all three analyzed axes, there exists a working range. Regarding the Z-axis (distance from the marker), the general rule is that the farther the camera is (with respect to the marker), the worse the estimated poses become. Marker dimensions must be carefully designed, anticipating and considering the conditions and distance at which the camera is expected to be during the relocalization phase. While a marker with a side length of 28 cm was used for this test, in a real-use scenario, it's impractical to use markers of such size. Despite the increased size, the estimated poses beyond a certain distance remain unreliable, as seen. Once the marker size is designed properly, based on the expected working range and desired performance, an appropriate Kallman filter must be developed.

Given the oscillations observed, it is impractical to develop a relocalization algorithm that does not incorporate smoothing and mitigation of measurement variations. To address Z-flipping and reduced accuracy near the range margins, it can be considered to construct a dedicated logic, on top of the Kallman filter, that essentially considers poses and measures only within a specific range/threshold. This would filter out poses near $X = 0$, $Y = 0$, and those near the FOV limit.

Future work Future analysis could involve a multi-variable comparative study, simultaneously evaluating multiple variables. In the current analysis, we focused on individual coordinates (X, Y, and Z). An evolution of this approach might entail analyzing these variables concurrently and exploring multiple axes to observe how different variables change from one respect to the other. However, a more suitable test bench would be necessary for such an investigation. One option could be mounting the camera on a robotic arm that moves in space at specific points, with known inclinations (*roll*, *pitch*, and *yaw*). This setup would allow estimation of both position and orientation (*pose*) coordinates, enabling comparison across 6 coordinates simultaneously.

Alternatively, experiments with fluorescent or reflective markers could be conducted to analyze their behavior in low or absent lighting conditions. Another approach would be to use traditional markers but with infrared cameras, such as the Picam with a wide-angle lens equipped with an IR sensor. Comparing performance across different infrared camera models and lighting conditions could yield valuable insights.

Further tests could be conducted using cameras more similar to the D435i in terms of hardware and optics/sensors. The wide-angle Picam used in this study had limitations due to the wide-angle lens and the fact that the infrared sensor is sensitive to visible light wavelengths, resulting in lower image quality compared to a standard RGB sensor with the same resolution. A potentially interesting camera for this comparison could be the Picam RGB with a 77° FOV (not like the used one that has a FOV of 160°), as it has a high-resolution sensor, lacks distortion from a wide-angle lens, and is designed to work with visible light wavelengths, potentially would have good results. From an engineering standpoint for drone integration, this camera could be particularly suitable. Despite being an additional camera compared to the D435i, it is compact and energy-efficient. This characteristic would enable placement in a different location on the drone, providing two distinct Points of View (POV).

Exploring the use of event-based cameras, which have been evolving in recent years, could be another interesting research idea. These cameras could employ different techniques to recognize fiducial markers, as proposed in the article [32].

Finally, a different approach could involve improving the markers themselves:

- Enhancing existing markers like Aruco by adding circles at the vertices to improve vertex estimation [33].
- Customizing a marker family based on specific needs, generating and based upon existing families [34].
- Developing an entirely new marker family based on color information rather than black and white [35].
- Creating libraries that combine information from multiple markers strategically positioned on different planes, compensating for errors and enhancing the accuracy of the estimated pose [36]. The idea is to use 5 Aruco markers: one central and four attached to each side of the central, with a specific inclination. This approach allows for diverse information from different markers on different planes. Any potential errors are then compensated, thereby enhancing the overall accuracy of the estimated pose.

Bibliography

[1] Jean-Claude Baraka Munyaka and Sarma Venkata Yadavalli. «INVENTORY MANAGEMENT CONCEPTS AND IMPLEMENTATIONS: A SYSTEMATIC REVIEW». In: *The South African Journal of Industrial Engineering* 33 (July 2022), pp. 15–36. DOI: `10.7166/33-2-2527`. URL: `https://sajie.journals.ac.za/pub/article/view/2527` (cit. on pp. 2, 3).

[2] «Ted Baker balance sheet error worse than feared as woes deepen». In: *The Guardian* (2020) (cit. on p. 2).

[3] «Mark lost his leg in a forklift accident, now he's warning other workers». In: *7 News Au* (2023) (cit. on p. 3).

[4] Digital Innovation. *Il mercato professionale dei droni: l'analisi a livello italiano e internazionale*. Tech. rep. Osservatori.net, Politecnico di Milano, 2021 (cit. on p. 5).

[5] In: *Business Inside* (2021) (cit. on p. 6).

[6] Netland T. Wawrla L. Maghazei O. «Applications of drones in warehouse operations.» In: *Whitepaper. ETH Zurich, D-MTEC, Chair of Production and Operations Management. Downloaded from www.pom.ethz.ch* (2019) (cit. on pp. 7–10).

[7] Brian Heater. «Verity raises 32$MasIKEAstoresdeployitsinventorydrones$». In: *Tech Crunch* (Mar. 2023) (cit. on p. 9).

[8] URL: `https://verity.net/` (cit. on p. 9).

[9] Christoph Roser. «How IKEA Uses Drones for Inventory Management». In: *AllAboutLean.com* (July 2022) (cit. on p. 9).

[10] Brianna Wessling. «100,000+ mobile robots shipped in 2021». In: *The Robot Report* (Mar. 2022) (cit. on p. 13).

[11] *AGV-AMR Market (4th Edition)*. Tech. rep. The logisitcs IQ, 2023 (cit. on p. 13).

[12] URL: `https://getfabric.com/` (cit. on p. 14).

[13] URL: `https://www.autostoresystem.com/` (cit. on p. 14).

[14] EU Drone Port. *JARUS Methodology for Evaluation of Automation.* Tech. rep. JARUS (cit. on pp. 21, 22).

[15] H. P. Morevec. *Towards automatic visual obstacle avoidance.* 5th Int. Joint Conf. Artif. Intell. (IJCAI), vol. 2, p. 584., 1977 (cit. on p. 44).

[16] C. G. Harris and J. M. Pike. *3D positional integration from image sequences.* Proc. Alvey Vis. Conf., Cambridge, U.K., 1987 (cit. on p. 44).

[17] TOMI WESTERLUND; JUKKA HEIKKONEN; HANNU TENHUNEN SHERIF A. S. MOHAMED MOHAMMAD-HASHEM HAGHBAYAN and JUHA PLOSILA. «A Survey on Odometry for Autonomous Navigation Systems». In: *IEEEAccess* (2019) (cit. on pp. 44, 47, 50, 51, 58, 71).

[18] N. Michael S. Shen Y. Mulgaonkar and V. Kumar. *Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV.* Proc. IEEE Int. Conf. Robot. Autom. (ICRA), 2014 (cit. on p. 51).

[19] Fellow Hugh Durrant-Whyte and Tim Bailey. «Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms». In: *IEEE* () (cit. on p. 56).

[20] Kobayashi H. Gezici S. Sahinoglu Z. and Poor H. V. «Ultra Wideband Geolocation». In: *John Wiley Sons, Ltd.* (2005) (cit. on p. 66).

[21] Ahmed. «Wheel Odometry Model for Differential Drive Robotics». In: *medium.com* (2023) (cit. on p. 69).

[22] Schouten Ramon. «Accuracy of Single Camera Pose Estimation with ArUco Fiducial Markers». Eindhoven University of Technology, 2022 (cit. on p. 72).

[23] Zhiqi Su; Gang Yang; Zhong Wang Yutao Wang Zongpeng Zheng and Yu Luo. «An Improved ArUco Marker for Monocular Vision Ranging». In: *IEEE* (2020) (cit. on pp. 73, 74).

[24] Andrej Babinec Eduard Mraz Jozef Rodina. «Using fiducial markers to improve localization of a drone». In: *IEEE* () (cit. on p. 74).

[25] F.J. Madrid-Cuevas S. Garrido-Jurado n R. Muñoz-Salinas and M.J. Marín-Jiménez. «Automatic generation and detection of highly reliable fiducial markers under occlusion». In: *Elsevier* (2014) (cit. on p. 91).

[26] URL: `https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html` (cit. on pp. 92, 94).

[27] URL: `https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html` (cit. on p. 93).

[28] Varlik Kilic. «Performance Improvement of a 3D Reconstruction Algorithm Using Single Camera Images, Master Thesis». Middle East Technical University, 2005 (cit. on p. 95).

[29] Andrej Babinec Eduard Mraz Jozef Rodina. «Using fiducial marker to improve localization of a drone». In: *IEEE* () (cit. on p. 105).

[30] URL: `https://github.com/opencv/opencv/issues/8813` (cit. on p. 105).

[31] URL: `https://optitag.io/blogs/news/designing-your-perfect-apriltag` (cit. on p. 117).

[32] MIGUEL A. OLIVARES-MENDEZ HAMID SARMADI RAFAEL MUÑOZ-SALINAS and RAFAEL MEDINA-CARNICER. «Detection of Binary Square Fiducial Markers Using an Event Camera». In: *IEEE* (2021) (cit. on p. 133).

[33] Zhiqi Su; Gang Yang; Zhong Wang Yutao Wang Zongpeng Zheng and Yu Luo. «An Improved ArUco Marker for Monocular Vision Ranging». In: *IEEE* (2020) (cit. on p. 133).

[34] SERGIO GARRIDO-JURADO DAVID JURADO-RODRÍGUEZ1 RAFAEL MUÑOZ-SALINAS and RAFAEL MEDINA-CARNICER. «Design, Detection, and Tracking of Customized Fiducial Markers». In: *IEEE* (2021) (cit. on p. 133).

[35] Hamid Nabati Laszlo Egri and Jia Yuan Yu Concordia University Montreal Canada. «RainbowTag: a Fiducial Marker System with a New Color Segmentation Algorithm». In: *IEEE* (2022) (cit. on p. 133).

[36] Aleš Vysocký; Jakub Mlotek; Petr Novák; Ivan Virgala; Marek Sukop Petr Oš˘cádal Dominik Heczko and Zdenko Bobovský. «Improved Pose Estimation of Aruco Tags Using a Novel 3D Placement Strategy». In: *https://www.mdpi.com/journal/ser* (2020) (cit. on p. 133).