ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF ELECTRICAL, ELECTRONIC, AND INFORMATION ENGINEERING “GUGLIELMO MARCONI” - DEI

SECOND CYCLE DEGREE IN AUTOMATION ENGINEERING

MOBILE MANIPULATION: FROM THE PROTOTYPE DEVELOPMENT TO THE INDUSTRIAL APPLICATIONS

Dissertation in Automation Software and Design Patterns

Supervisor

Prof. Gianluca Palli

Co-Supervisor

Alberto Pepe

Defended by

Domenico Petrella

Graduation Session March 2025

Academic Year 2023/2024

TABLE OF CONTENTS

INTRODUCTION

Looking at the history of robotics development, it is observable that the increase in the functionality of robots causes a rise in the number of their possible applications in various fields of human activity. The creation of robots was preceded by the idea of replacing a person with hard work, and the physical capabilities of the human body served as a model for them. Robotics can be categorized into three main branches.

The first is industrial robotics with reprogrammable and multi-purpose manipulators made by three or more axes, which can only be fixed in the workspace [1]. Industrial robots, especially robot manipulators, have been widely and successfully used in conventional production processes due to their high endurance, speed, and precision in structured environments [2]. Many traditional industrial sectors have been based on the serial production line for decades to optimize results and be increasingly efficient in the manufacturing of large batches of identical products [3].

As a progressive stage in developing industrial robots, the second strategy is collaborative robotics. This is a new step in developing industrial robots, assuming they will interact closely with humans, guaranteeing safety. These robots are equipped with a wide range of sensors and vision systems. For example, if a person enters the workspace of a moving robot, it must either stop or change its trajectory.

The actual trend is service robotics, represented by mobile autonomous/semi-autonomous robots, including collaborative robot manipulators, used in various fields of human activity. These robots perform helpful work for people and equipment [1]. In recent years, there has been a significant shift in market needs. Product personalization and differentiation have become key factors when purchasing a wide variety of non-basic products. Adaptating to this new production, recently known as "*mass customization*", is the key to keeping manufacturing companies competitive in these sectors, with smaller but more variable production volumes. Traditional robotics does not respond to the current market demand for changing products with small production batches. The robustness and efficiency of the serial production model is highly compromised by the need to perform changes in production equipment [3].

Todays, robots go beyond their limits and become flexible, mobile and smarter. As part of Industry 4.0, robots have become the driving force of automation as it has never been before [1]. Industry 4.0 seeks to develop smart factories that are highly efficient, adaptable and responsive to market changes. To satisfy the dynamic requirements of the factory of the future, the next generation of robots and its accompanying technologies will play a major role. Specifically, the cutting-edge sensor technologies will be crucial for future advancements in robotics. These sensors not only enhance robot understanding of their surroundings but also ensure safe and efficient communication between humans and robots [4].

The rise of autonomous systems and robotics, especially collaborative robots, is opening new market possibilities. A new trend for flexible and collaborative robotics is spreading in the industry in the form of autonomous industrial mobile manipulators. These hybrid systems combine two fundamental robotic skills: mobility in an environment and the manipulation of objects [5]. The ability to do both simultaneously opens numerous applications in diverse areas, including manufacturing, logistics, home automation, and health care. Such applications typically require complex manipulation. They also demand navigation in large spaces, possibly in cooperation with human beings and other robotic systems. Merging mobility and manipulation, mobile manipulation systems need to overcome some of the most difficult challenges in robotics: they must be able to continuously adapt and improve their performance [6]. Among the various kinds of robots, mobile platform with manipulator presents a promising solution for automation, since they could move flexibly and pick stuff easily, which is very suitable to work in factory for pick-and-place tasks [7].

In the aforementioned context, this thesis aims at analyzing a real Mobile Manipulator system. Throughout this work, an in-depth analysis of a real MoMa (Mobile Manipulator) prototype - consisting of an autonomous mobile robot (AMR) equipped with a collaborative robot (cobot) - is conducted.

The first section presents a state-of-the-art review, analyzing the development of mobile manipulation among the years, analyzing the history, the architecture and the future impact on industrial applications. Then, mobile robots and cobots are presented, emphasizing the concepts of localization, navigation and obstacle avoidance for AMRs, while highlighting the human-robot interaction and collision avoidance for manipulators. At the end of this section, some theoretical aspects related to the implementation of a vision system are reported, focusing on the visual servoing technique.

In the second section, the work presents an investigation and definition of the comprehensive hardware architecture of the prototype. This includes identifying the technical requirements, communication protocols and system interfaces to enable efficient interoperability among AMR and cobot.

In the third section, the focus is centered the software development environments for both the cobot and the AMR to program coordinated missions, ensuring seamless collaboration between the two robotic systems. Even the technical details of the camera are reported, since it plays a crucial role in the overall application. The robots must be integrated in such a way that the entire system must behave as a unified product. For this purpose, in conclusion of this section, a detailed representation and exploration of internal connection is carried out, addressing the role of every component in the overall architecture.

The contribution of this work reported in the fourth section aims to validate the system, performing tests necessary for a deep undertaking of the overall functioning. At the beginning, some tests are focused on the validation of a 2D camera-based marker detection system. This solution has the goal to identify markers and establish a 3D reference frame, allowing the cobot to align its movements accurately within the workspace. This approach aims at decoupling the arm trajectory precision from the AMR, addressing inherent limitations in the AMR accuracy. Then, a proper estimation of the accuracy is executed, reporting some experimental data that comes out from the alignment operation. Eventually, the last test has the objective of presenting the prototype as a unified system capable of performing tasks in a real industrial-like application scenario, demonstrating the system capabilities and performance under operational conditions.

This cutting-edge robotics scenario integrating manipulation capabilities with mobile robots is expected to contribute to the development of innovative industrial solutions by extending the cobot workspace through the mobility of the AMR.

CHAPTER 1 – STATE OF ART

Mobile manipulators will play a crucial role in the transition to Industry 4.0, as they offer an easy way to reconfigure the assembly line by self-reorganization, providing more options for cooperation with humans. Shortly, mobile manipulators may become an integral component of smart factories, workshops and highly modular assembly lines [8].

Compared with traditional industrial robots, mobile manipulators can transfer their position, providing highly flexible services and dexterity in any environment. Therefore, they can be widely used in different scenarios, including industries, living apartments, orchards, precision agriculture [9]. They can be designed and manufactured for various industrial purposes, including polishing, sanding, painting, assembly, packaging, logistics and other challenges of modern processes. The growth in e-commerce also led to the development of these robots in warehouses, which are used for automated storage or product retrieval. Furthermore, they are involved in military and rescue operations, since they could enter more hazardous environments to fulfill dangerous tasks, such as defusing bombs and the remote inspections.

1.1OVERALL ARCHITECTURE

Mobile manipulation is a widespread term used to refer to robotic systems consisting of a robotic arm rigidly mounted on a mobile platform. This layout allows the robot to perform tasks that require both locomotion and manipulation abilities. The concept of mobile manipulation dates back to 1984 when the Mobile Robot (MORO) was introduced. Then, a lot of research and development has carried out and now the mobile manipulation technology is on the edge of its breakthrough within different domains [10].

Due to its complex design, mobile manipulation requires a close relationship between scientific research and the needs of industries in multiple innovative fields: perception, navigation, path and grasp planning, control, error recovery, human–robot interaction and robotic hardware development. The challenge in mobile manipulation is to obtain an integrated system that can combine a large variety of hardware and software components to increase the range of tasks that a robot can perform [6].

Another barrier is balancing the relationship between modularity and integration. On the one hand, modularity decomposes the whole complexity into several subsystems, which would be easier to build up. On the other, the synergies of the integrating systems should be considered to ensure that the incorporation of the components would make the whole system work more effectively [5].

1.1.1 HARDWARE STRUCTURE

The hardware of a typical Mobile Manipulator (MoMa) contains four main subsystems: a manipulator necessary for tasks execution, a mobile platform required for the movement, a set of sensors to perceive the environment and the end effector. The mobile platform is used for navigation and localization. Compared to traditional production line robots, the ability to change its workplace represents a great differentiating leap since there is no need to install a robot for each work cell [7].

The mobility of these systems can take multiple forms, depending on their environment: air/space (drones, planes, helicopters, and satellites), water (ships and submarines), and land (wheeled and legged robots). For ground-based mobility, different solutions have been introduced [6].

A mobile manipulator with a railway guided vehicle, robot can only execute the manipulation tasks in a limited space, since it can only move in a single dimension in a restricted area (*Figure 1*). To overcome these limitations, a more flexible solution is represented by wheeled vehicles which can move with unrestricted motion in any direction, avoiding the obstacles in a limited or unknown environment [9] (*Figure 1*).

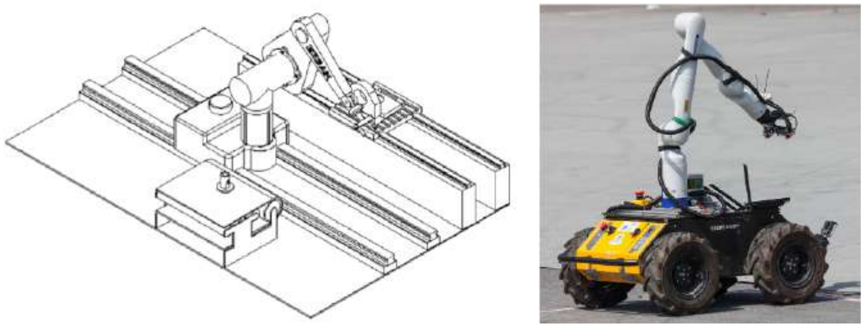


Figure 1 – On the left, a railway guided mobile manipulator [29]. On the right, a wheeled mobile manipulator [8].

Nowadays, there exist two main kinds of manipulator. The first one is large, having extensive applications in industry for which they are pre-programmed. This model must be fenced because its working velocity is relatively high, which is dangerous for humans. The second one has a smaller size and this makes it compatible with human interaction. In general, its work velocity and payload are quite low, it could be mounted into a mobile platform to do more complex and flexible tasks.

Different kinds of sensor should be used to obtain the desired information from the environment. Some of them are used for localization, such as encoders, GPS, laser scanner while others, like cameras, are used for identification. Force/torque sensor can be mounted on end-effector, disabling the system if the force detected exceeds the safety threshold. High resolution sensors could make the whole system more flexible and accurate for navigation and collision avoidance. End-effectors are always used for gripping or performing assignments. The shape of end-effectors changes according to the task [7].

Endurance is another key issue in mobile manipulators. While static robots are permanently connected to power, mobile robots rely on limited life batteries to operate, which should last enough to not interfere with the production process. Thus, a set of batteries able to work for a whole 8 hours turn should be provided, with the possibility of doing opportunity charging while not performing specific tasks. Batteries are big and heavy components; they need to be placed in the lower part of the system to maintain the overall stability of the platform. Even a pneumatic system is usually present in those platforms. A small compressor could be installed to feed pneumatic tools fixed on the robotic arm and other secondary applications with low flow demand, like tool exchangers [3].

1.1.2 SOFTWARE CONFIGURATION

Software part involves multiple modules, combining disciplines as programming, control, interaction and manipulation. In general, different tasks need to adopt suitable software strategies and they always need to be divided into several steps because of the complexity of their architectures. Path and motion planning contains two steps: motion planning for manipulator and path planning for mobile platform. The common planning algorithms could be divided into preprogrammed and real-time programmed, which depends on knowledge of environment map. Nowadays, the trend moves towards more flexible mechanisms, where the system could update the map simultaneously thereby increasing the effectiveness. In general, there are three kinds of control: manual, semiautonomous and fully autonomous. Manual control is safer because of directly handling of the errors, even if it requires the intervention of specialized human. Semi-autonomous control strategy combines the advantages of both manual and autonomous control. To allow the system to learn and act independently, autonomous control is the best way to achieve fully intelligent industry, with high decision cost [7]. The Figure 2 shows how the control software is linked to different modules composing the conventional architecture of a mobile manipulator.

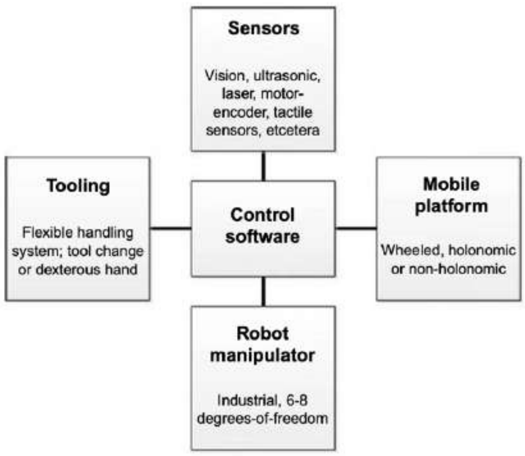


Figure 2 – The conventional architecture of a mobile manipulator [10].

1.1.3 PRECISION AND ACCURACY

Mobile manipulation is a complex field. Mobility introduces additional pose uncertainty to the manipulation problem while introducing constraints to the navigation issue. Versatile robotic systems must be equipped with many actuators and sensors, resulting in high-dimensional state spaces for planning and control. The ability to locomote, the required adaption in task execution and the use of multiple sensors and actuators make arising problems due to the uncertainty of sensing and actuation [6].

In a work cell containing a stationary industrial robot, the precise location of objects is typically known, thus collision free paths and manipulation can be offline programmed. In mobile manipulation, the precise location of the robot system in a scene is not known, as localization techniques are only accurate down to a few centimeters [10].

After the global autonomous navigation, the robot is driven to the proximity of the operation zone. Here, it is necessary to perform an accurate positioning to ensure a correct placement of the manipulator in the workspace. Some approaches use cameras to identify fixed markers in the scene. The reference goal is recorded in a previous calibration process. Then, a visual servoing system based on a proportional control maintains and ensures, with a certain accuracy, the position of the robot with respect to the marker [5]. Since the transformation between the camera and the cobot is rigid, it is necessary to guarantee that the marker fall within the field of view of the camera. Thus, the ability to recognize the marker 6D pose, to estimate 3D position and 3D orientation, is essential to determine the pose of the manipulator with respect to the map of the workstation and grab objects successfully. With this strategy, standard navigation techniques will be able to achieve adequate positioning accuracy.

1.1.4 INDUSTRIAL APPLICATION

Mobile manipulators have most relevant applications in industry areas, where the sales have increased rapidly in recent years. They can carry out their missions by navigating between workstations and performing diverse manufacturing tasks. Even if the mobile manipulation market remains a niche, it is predictable that sales would have a considerable increase, and some key technology would get breakthrough in industry 4.0.

Logistics is the area that calls for vast mobile manipulators because of the high volume of transferring tasks. The operation environments are always stable such as a factory or warehouse and extensive research regarding object transferring capabilities has already been conducted. Manufacturing is another area in which this technology has been shown to have high utility. Furthermore, the system could be used in processing/reprocessing line as material transferring tool. Compared to stationary manipulators, mobile manipulators are more flexible and effective, which means that they could make more production and reduce cost in industry. Assembly is an essential process in automotive industry which currently require human labor to transfer large or heavy components: some specific mobile manipulators are designed for that [7].

1.2 MOBILE ROBOT

A mobile robot is an automatic machine capable of moving within its environment, either autonomously or teleoperated by humans. It has a wide range of use, including indoor applications like service robotics for hospital assistance in transporting medicines and sanitary materials and outdoor applications like underwear robotics for water monitoring in deep sea, space robotics for ground inspections or military operations for mine clearance. In recent years, automated operations have also been employed in the agricultural field for fertilizers and pesticides management, introducing visual systems for color recognition to determine when some fruits are ripe and ready for collection.

Automatic Guided Vehicles (AGVs) were one of the earliest applications for mobile robots. The first AGVs were deployed in the 1950s to transport materials in large facilities and warehouses. Nowadays, AGVs are widely used for end-of-production-line management in the fields of food and beverage, pharmaceutical, automotive, transporting pallets of stored products to truck loading areas. Typically, they can pull carts or automatically pickup and drop loads off from various heights as mobile forklifts [11]. Some examples are depicted below.

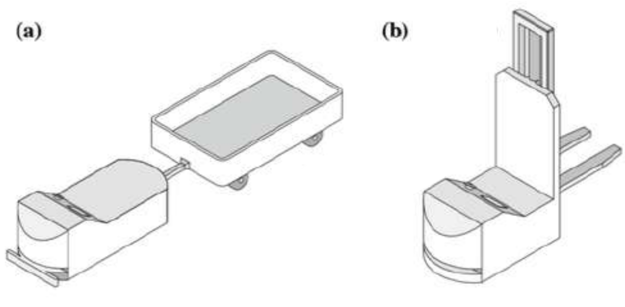


Figure 3 – a) AGV pulling a cart, b) mobile forklift [11].

Generally, as per definition, AGVs are conveyor systems guided by some hardware placed in the surrounding environment. In contrast, an Autonomous Mobile Robot (AMR) can move around autonomously, navigating through sensors. The latter better adapt to real world application, allowing dynamic customer specific modifications. Anyway, up to now, there is no absolutely clear distinction established [12].

Compared to fixed automation, the usage of mobile robots has more advantages in terms of cost, flexibility, installation, reducing human interaction which drastically improve the overall efficiency of the workplace. The deployment of the mobile robots in the industry will increase considerable in the future. According to *Research And Markets*, the market for autonomous mobile robots will rise exponentially by 35% per year and will reach around $13.2 billion dollars at the end of 2026 [13].

The navigation is typically a complex task consisting of localization, path planning and motion control. Localization denotes robot's ability to establish its own position and orientation within the global coordinate frame. Autonomous path planning represents determination of a collision-free path for a robot between start and goal positions, avoiding obstacles cluttered in a workspace. Motion control must guarantee execution of movement along the planned path with simultaneous obstacle avoidance. For the traditional autonomous navigation approach, the robot collects information of the environment through sensors and constructs a static map representation. These sensors play a key role in navigation, helping the robot to perceive its surroundings in real time, maintaining the control over its position and orientation [11].

1.2.1 STRUCTURE

The locomotion of mobile robots, in most of the ground-based cases, is ensured by the presence of wheels, which differs according to the application. A minimum of three wheels is sufficient to guarantee the static stability of the vehicle. The most basic type is the fixed wheel, which can only rotate around its own axis and does not allow lateral movement. A more flexible option is represented by the centered adjustable wheel, which can spin around a vertical axis passing through its center, making it useful for controlling direction changes. The caster wheel, on the other hand, can freely rotate about different vertical axes, but they are only passively guided by the rest of the system. A more advanced solution is the Mecanum wheel, which consists of angled rollers that enable movement in multiple directions. This design makes them particularly suitable for applications requiring high maneuverability. The combination of different wheel rotations allows a mobile robot to perform various types of movement, including translation and rotation, making it highly adaptable to complex environments.

1.2.2 KINEMATIC MODEL

The choice of wheel type directly influences the robot's mobility and kinematic constraints. The configuration space has dimensions equal to the number of parameters needed to uniquely describe the configuration of a mobile robot. The robot does not involve movement in the Z-axis direction; therefore, it can be modeled in a 2D environment representation. Each wheel introduces in the system a non-holonomic constraint since normal translations to rolling direction are not allowed. The kinematic model of a two actuated wheels mobile robot, depicted in *Figure 4*, can be obtained by utilizing the speed difference between the two wheels to control the rotation and translation [14].

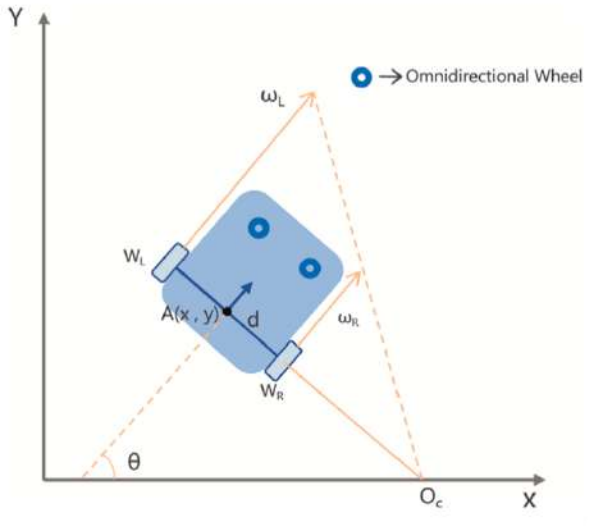


Figure 4 – Graphical representation of the differential motion model of a two actuated wheels mobile robot. The Oc is instantaneous center of rotation (ICR) and ωL, ωR are angular velocity of the robot [14].

The robot's pose is represented as q = [x y θ] ^{T}, where *(x, y)* denotes the position in two-dimensional coordinates and θ defines the pose angle. Assuming the left and right wheel velocities of the robot are ω_L and ω_R, respectively, and the distance between the wheels is *d*, the linear velocity of the left and right wheels of the robot can be calculated as:

$$\boldsymbol{v_L = \frac{d}{2} \times \omega_L}$$

$$\boldsymbol{v_R = \frac{d}{2} \times \omega_R}$$

The overall velocities result:

$$\boldsymbol{v} = \frac{\boldsymbol{v_R} + \boldsymbol{v_L}}{\mathbf{2}}$$

$$\boldsymbol{\omega} = \frac{\boldsymbol{v_R} - \boldsymbol{v_L}}{\boldsymbol{d}}$$

where v represents the linear velocity of the contact point between the wheel and the ground and ω is the wheel angular velocity [14]. By adjusting these variables, it is possible to modify the robot configuration, whatever navigation algorithm adopted. The pose is described as follows:

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{y}} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}\cos\boldsymbol{\theta} \\ \boldsymbol{v}\sin\boldsymbol{\theta} \\ \boldsymbol{\omega} \end{bmatrix}$$

1.2.3 LOCALIZATION

The important difference between a manipulator and a mobile robot is in position estimation. A manipulator has a fixed base and by measuring robot joint positions and knowing its kinematic model it is possible to determine the pose of its end-effector. A mobile robot can move through the environment and there is no direct way for measuring its position and orientation. A general solution is to estimate the robot position and orientation through integration of the velocity over the time. However, more accurate and more complex approaches are typically required. If the map of the environment is known in advance, mobile robot paths can be preplanned. This is specifically useful when the environment is relatively static and robust operations are required, such as in industrial applications. When the workspace or the tasks change frequently it is typically better to plan dynamically. Often a trade-off is required between preplanning and dynamic generation of paths, based on sensor information and recognition of features in the environment [11].

For this aim, LIDAR sensor is frequently used: it works independently or coupled with other sensors like GPS or motors encoders. This technology can be adopted for mapping the local environment to locate and identify the landmarks position, which in general is referred to as SLAM (Simultaneous Localization and Mapping). With the help of this strategy, mobile robot automatically corrects its position and orientation. SLAM employs a technique to gather information about the environment to estimate the precise positioning and orientation. SLAM algorithms can be used in various applications, including self-driving vehicles, mobile robots and drones. One of the main challenges is achieving high accuracy and precision in the map. This is particularly important in applications where the robot needs to navigate through tight spaces or perform precise tasks.

The odometry is a technique to estimate the pose of a wheeled vehicle based on information from sensors, measuring the space covered by the wheels and their steering angles by tracking their rotation at runtime. Errors accumulate over the time due to wheel slip or incorrect kinematic parameters calibration, reducing the overall system accuracy if distances increase. Usually, SLAM and odometry techniques still face major issues in algorithm robustness, both in static scenarios and real-world dynamic environments. The algorithms can find very challenging deal with memory usage and map storage. SLAM algorithms can be computationally intensive, which can be an issue for robots with limited processing power or for real-time applications, where the robot needs to constantly update its map and location estimate [16].

1.2.4 NAVIGATION

Mobile robots often operate in unknown and unstructured environments, they need to self-localize and then control their motion through that environment [11]. The navigation is an essential issue in the field of robotics. Path planning has been considered as the most-common problem for robot navigation, robots must move along a collision-free path from starting position to goal position minimizing the total cost of the associated path. The path determination for mobile robot can be classified as static or dynamic. For static path, vehicle uses pre-defined path between origin and destination point, with the help of some additional hardware components as embedded wires and magnetic tapes. In dynamic path, vehicle behaves autonomously to determine path, it only knows its destination. Internal navigation system is used by vehicle to reach its destination. Depending on the nature of environment, path planning can be divided into static and dynamic. If obstacles are stationary, it is referred as static path planning while if obstacles change their position and orientation with respect to time, then it is referred as dynamic path planning [15]. Reliable navigation can be achieved with various strategies developed over the time. For global navigation, prior knowledge of the environment is required, a concept known as offline path planning. Examples of such methods include Dijkstra's algorithm, A* and the potential field method.

The Dijkstra algorithm, also known as graph search method, is considered the simplest approach for finding a path for a robot, solving the optimal path problem by producing shortest route. It is a well-defined, effective and computationally efficient method for identifying a nonobstructive path made with a line that allow the robot to easily reach the target, iteratively computing the shortest distance to the endpoint. The A* is a search algorithm that can also be used for path-finding. The algorithm continuously searches for unexplored location in graph. This algorithm is the popular for path finding in games. The algorithm based on Potential Field depends on two forces: attractive and repulsive. The basic idea of is to fill the robot environment with the potential field in which goal produces the attractive force towards robot and obstacles produce repulsive force, which is inversely proportional to the distance between the robot and the obstacles. The robot then should calculate potential fields at runtime and move according to forces, travelling from high potential to low potential. The major problem is that robot may trap is local/global minima problem, the robot can get stuck at point where both forces cancel out the effect of each other and does not allow the robot to move further or even backward [15].

1.1.5 PERFORMANCES AND SAFETY

Path planning involves a lot of factors to be considered for producing the best possible results such as path length, stability, efficiency and safety. There are various metrics that can be used to evaluate the performance of a navigation system, but none of them is able to indicate the quality of the whole system. Therefore, it is necessary to use a combination of different indexes quantifying different aspects of the system. Navigation performance metrics can be classified according to security and smoothness of the trajectory toward the goal [17]. The robot safety along a trajectory is evaluated by considering the distance between the vehicle and the obstacles in its path.

LAW	Machinery Directive 2006/42/EG National Law for the Protection of Health and Safety (ASchG 1995)			
ISO STANDARD	Type A	EN ISO 12100 ✓ Safety of machinery- Integrated manufacturing systems- Risk assesment and risk reduction		
	Type B	EN ISO 11161 ✓ Safety of machinery- Integrated manufacturing systems- Basic requirements	EN ISO 13849 ✓ Safety of machinery- Safety-related parts of control systems	
	Type C	ISO 10218 ✓ Robots and robotic devices- Safety requirements for industrial robots	EN 1525 Safety of industrial trucks- Driverless trucks and their systems	EN ISO 3691-4 ✓ Industrial trucks- Driverless industrial trucks and their systems (draft)
IEC STANDARD	EN 61508 Functional safety of e/e/pe safety-related systems	EN 62061 ✓ Safety of machinery- Functional safety of safety-related e/e/pe control systems		
GUIDELINE	ISO TS 15066 Robots and robotic devices- Collaborative robots	VDI 2510 Automated guided vehicle systems	VDI 2710 Interdisciplinary design of automated guided vehicle systems	

Figure 5 - Overview over relevant laws, standards and guidelines for mobile robot systems (automated guided vehicles and mobile manipulators). Standards highlighted by a green tick are harmonized with the machinery directive.

1.2 COBOT

In the 1960s, the first industrial robots were developed. In the meantime, robot-based automation solutions have become an indispensable part of today's industry. The further development and implementation of robots determines the design, efficiency, optimization and rationalization of actual work and production processes. Therefore, technological progress represents one of the most important developments in economic ecosystems, industry and the working world. Between 2011 and 2021, the number of industrial robots worldwide increased by about 200 percent [18].

Industrial robots are programmable and autonomous machines comprising electronic, electrical, and mechanical components. They can execute a complex set of tasks that require repetitive and monotonous operations. These robots are massive, inflexible and are usually installed to perform tasks that may be hazardous for human, such as transporting heavy loads in factories [19]. Industrial robots are usually installed in spatially separated work areas behind protective safeguards so that there is no direct cooperation between humans and robots [18]. These safeguards may be physical barriers (fences), or sensor-based systems (safety laser scanners) that result in protective stops of the equipment if a worker crosses a given boundary delineating what is known as the "*safeguarded space*" [20].

The challenge of developing flexible solutions where production lines can be quickly replanned, adapted and structured for new or slightly changed products is still an important open problem [21]. More flexible machines can be obtained by integrating sensors and tools, making them to adapt to a variety of production tasks, requirements, and situations. Current research activities in industrial robotics are increasingly focusing on the collaboration between humans and robots. especially in the field of human-centered production [18].

Industry 4.0, the latest and most advanced concept of industrial revolution, was coined in Germany in 2011. Industry 4.0 uses digitalization and networked production, incorporating IoT, cyber-physical systems and cloud computing to create "*Smart Factories*" [19]. The idea of the Industry 4.0 paradigm is that an industrial robot can serve as a cooperative and auxiliary tool for humans in production [1]. Although the concept of collaborative robots predates Industry 4.0, they have become increasingly relevant to the production and manufacturing industry with the advent of this latest revolution [19].

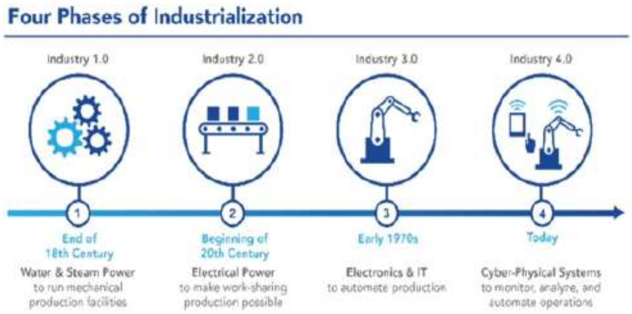


Figure 6 – The history of the industrialization phases among the years [19].

Collaborative robots, commonly known as cobots, are transforming the way humans and robots collaborate [19].

"Cobot" is a term used to describe a robot that is designed to assist humans in a specific task or allow humans and robots to work simultaneously in the same workspace, with specific design factors that make it safer for cooperative work [20]. The need for enhanced productivity and efficiency in industries, including manufacturing, logistics and healthcare, has fueled the development of cobots [19]. The main differences between traditional and collaborative robotics are reported in *Table 1*.

	Conventional robot	Cobot
Purpose	Perform tasks automatically	Collaborate with humans
Programming	Typically pre-programmed or scripted	Easily programmed by humans
Safety	Require safety measures and barriers to protect humans	Designed to work alongside humans without endangering them
Flexibility	Designed to perform a specific group of tasks	Can perform a wide range of tasks
Cost	Typically expensive	Less expensive than conventional robots
Complexity	More complex to program and operate	Simple to program and operate
Payload	High payload capacity	Lower payload capacity
Accuracy	Lower precision	High precision
Size	Bulky and Space consuming	Smaller and dense
Application	Manufacturing and assembly lines	Small-scale manufacturing, research and development

Table 1 – Differences between industrial robots and cobots [19].

Cobots are distinct from conventional industrial robots as they are intended to operate efficiently in conjunction with human workers, providing greater flexibility and adaptability in the workplace, guaranteeing safety without the use of safeguards. These robots are significantly lighter than traditional industrial robots, enabling greater mobility and ease of movement within the workspace where they are installed. One of the advantages that cobots offer over industrial robots is their flexibility, as they can be used to perform multiple tasks, making them highly adaptable to changing work requirements [19]. Due to these construction characteristics, direct physical interaction in the workspace between humans and robots during the execution of a production process becomes possible [18].

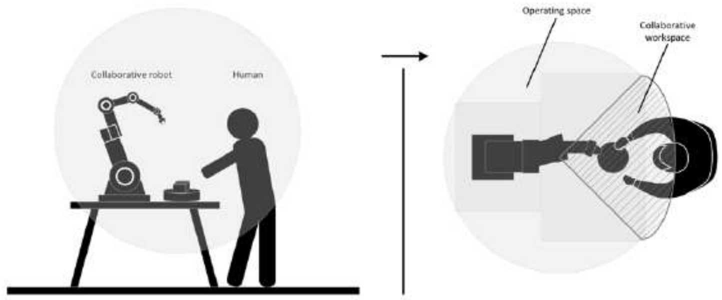


Figure 7 - Operating space including collaborative space of human-robot interaction [1].

They are programmed to perform a range of tasks, such as assembly, welding, packaging, and inspection. Cobots have a range of sensors on board and technologies that allow them to detect and avoid collisions with human workers and the environment [19].
In the collaborative environment, the human worker is considered as a kind of mechanical colleague from the position of a collaborative robot: the task of the robot is to help and assist the operator in achieving the goal in a comfortable environment [1].

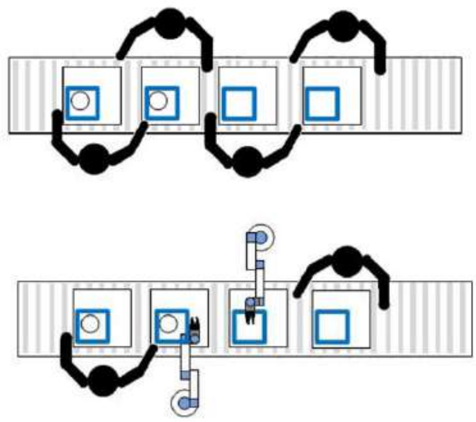


Figure 8 - Collaborative robot systems free workers to perform higher-value-added tasks [20].

1.2.1 COLLABORATION AND INTERACTION

The robot usage has changed in the last few years, from an idea in which robots work with complete autonomy to a scenario where robots collaborate with human beings. This brings together the best of each partner, by combining coordination, dexterity and cognitive capabilities of humans with the robots' accuracy, agility and ability to produce repetitive work [22].

Studies in human-robot collaboration (HRC) specifically explore the feasibility of use of robotic platforms at a very close proximity with the user for joint tasks [23]. The deployment of collaborative robots goes beyond the purely technological perspective and leads to profound changes in the design production processes in a company [18].

Implementing a collaborative robot cell confers several advantages as compared to traditional industrial robot cells. The advantages include cost reduction due to the elimination of safeguards, workspace optimization due to smaller footprint of cobots and partially automating tasks that still require dexterity, flexibility, and creative problem-solving of a human worker. Some common applications include assembly tasks, pick-and-place operations and quality control inspection [20]. HRC can be divided into four categories, according to the interaction with humans:

- Coexistence: robots work independently in a shared environment with humans, without any interaction or coordination of tasks. There is no direct sharing of the workspace or contact between the humans and robots.
- Synchronization: the work areas of humans and robots overlap, with both working on the same task. However, the work in the collaboration space takes place with temporal separation, without any dependency or contact. In some cases, a human operator and a collaborative robot can work simultaneously on different production processes at the same workpiece.
- Cooperation: humans and robots work on a common goal in a shared workspace. The operations of the human and the cobot are time-dependent, with the cobot handling more time-consuming tasks, which can also improve the operator's working conditions and reduce idle time.
- Collaboration: humans and robots work simultaneously on the same workpiece, executing a complex work task with direct interaction and complete dependencies between them. In these scenarios, the human performs tasks requiring dexterity or

decision-making, while the cobot handles repetitive, precise, dangerous, or force-intensive tasks [18][19].

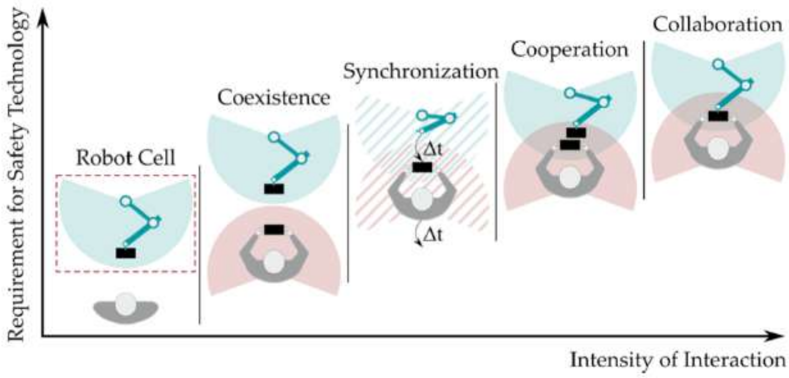


Figure 9 - Types of collaboration in HRC scenarios [18].

1.2.2 COLLISION AVOIDANCE AND SAFETY

In 1942 Isaac Asimov published the science fiction novel "*I, Robot*", where the three laws of robotics were introduced. First rule stated that "A robot may not injure a human being or, through inaction, allow a human being to come to harm".

This emphasizes how crucial safety is with the evolution of cobots, since robots are now capable of working alongside humans and performing tasks in proximity, eliminating physical separation [19]. The increasing demand by industry for collaborative robot-based solutions makes the need for advanced collision avoidance strategies more visible [22].

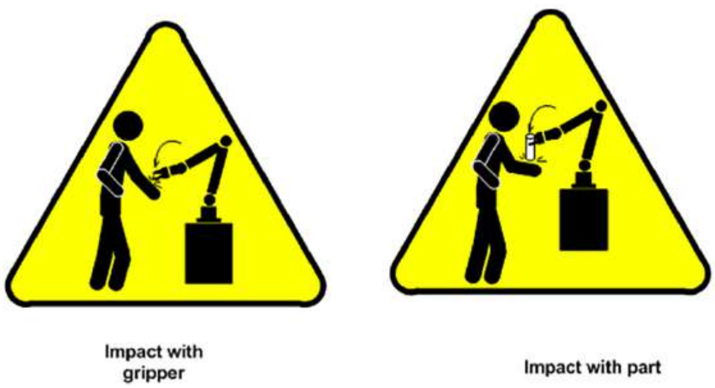


Figure 10 - Warning graphics illustrating two potential hazards of a collaborative robot system [20].

To have them working safely alongside with humans, robots need to be provided with biological-like reflexes, allowing them to circumvent obstacles and avoid collisions. This is extremely important to give robots more autonomy and minimum need for human intervention, especially when robots are operating in dynamic environment and interacting with human co-workers. Although it is widely claimed that collaborative robots are safe, this belief leads to the misconception that a risk assessment is not required when designing a collaborative robot application. The robot is not safe out of fences but rather has built-in features that provide alternative forms of protection [20].

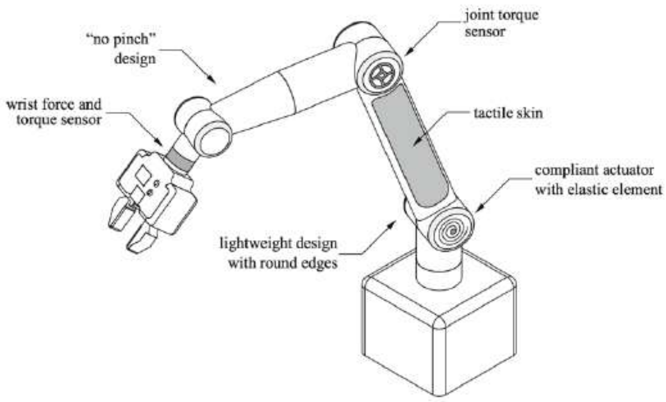


Figure 11 – Ergonomics and design features of a collaborative robot [11].

Collision avoidance is an important factor for human-robot safety. Two major problems in on-line human–robot collision avoidance can be identified. The first is related with the reliable acquisition of the human pose in unstructured environments. The second is due to the difficulty in achieving smooth continuous robot motion while generating collision avoidance paths. Hypothetical attraction and repulsion vectors attract the robot towards the target while repelling it away from obstacles [22].

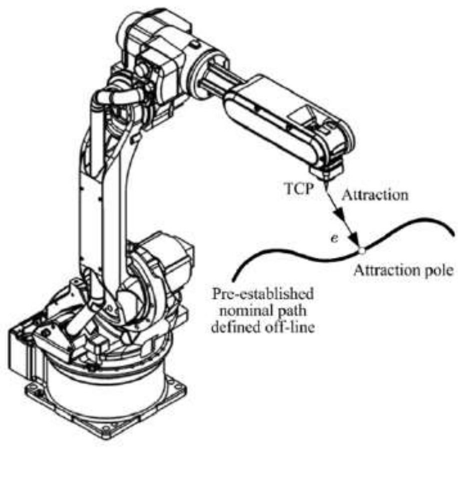


Figure 12 - The nominal path curve defined off-line, the attraction pole, and the error vector [22].

The idea of contact avoidance focuses on preemptively addressing the mechanical risks to operators by implementing preventive methods and systems to avoid hazardous contact. The idea of contact detection and mitigation is focused on ensuring the safety of operators in terms of mechanical risk by reducing the energy exchanged during unexpected or accidental contact between humans and robots. This is accomplished through the implementation of systems aimed at detecting and mitigating collisions through technologies based on compliance control, force feedback and proximity sensors [19]. There are four different types of collaborative operation defined in the robot safety standard, reported in *Table 2* [20]:

	Speed	Torques	Operator controls	Technique
Safety-rated monitored stop	Zero while operator is in collaborative workspace	Gravity and load compensation only	None while operator is in collaborative workspace	No motion in the presence of the operator
Hand guiding	Safety-rated monitored speed	As by direct operator input	Emergency stop, enabling device, motion input	Motion only by direct operator input
Speed and separation monitoring	Safety-rated monitored speed	As required to maintain min. separation distance and to execute the application	None while operator is in collaborative workspace	Prevented contact between the robot system and the operator
Power and force limiting	Max. determined speed to limit impact forces	Max. determined torque to limit static forces	As required by application	Robot cannot impart excessive force (by design or control)

Table 2 - Types of collaborative operations [11].

1.2CAMERA

Visual sensing is an important feature of an intelligent robotic system [24]. Consider if the robot could see the object and could use that information to guide the end-effector toward the object. This is what humans call hand-eye coordination and what in robotics is called vision-based control: the use of information from one or more cameras to plan a trajectory and guide a robot to achieve a task [25]. Vision plays a key role in a robotic system, as it can be used to obtain geometrical and qualitative information on the environment where the robot operates. Such information may be employed by the control system at different levels, for the task planning and for feedback control. In particular, the control based on feedback of visual measurements is termed visual servoing [26].

1.3.1 CONFIGURATION

A visual system may consist of only one or more cameras. The human capability of perceiving objects in three dimensions relies on the fact that the brain receives the same images from two eyes, observing the same scene from slightly different angles. The 3D vision can be achieved even with one camera, providing two images of the same object taken from two different perspectives. In many applications, mono-camera systems are often preferred because they are cheaper and easier to calibrate, although characterized by lower accuracy. For mono-camera systems there are two options: the fixed configuration, often referred to as eye-to-hand, where the camera is mounted in a fixed location, and the mobile configuration, or eye-in-hand, with the camera attached to the robot [26].

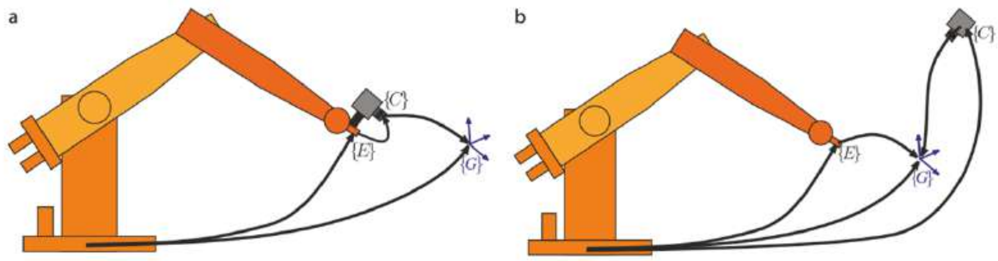


Figure 13 - Visual servo configurations: a) eye-in-hand, b) eye-to-hand, with relevant coordinate frames: world, end-effector {E},camera {C} and goal {G} [25].

In the eye-to-hand configuration, the visual system observes the objects to be manipulated by a fixed pose with respect to the base frame of the manipulator. The advantage is that the camera field of view does not change during the execution of the task, implying that the accuracy of such measurements is constant. However, in certain applications, it is difficult to prevent that the manipulator, moving in the camera field of view, hides the objects. In the eye-in-hand configuration, the camera is placed on the manipulator, usually mounted on the end effector, where typically can only observe the object. In this case, the camera field of view changes significantly during the motion and this produces considerable variability in the accuracy of measurements, with the advantage that occlusions are absent [26]. A popular configuration widely used in a variety of robotic applications is to mount a camera on the robot manipulator hand. Before performing a measurement task using such a system, both the camera and the robot need to be calibrated [24].

1.3.2 CALIBRATION

The geometric relationships between 2D views of a scene and the corresponding 3D space serve as basis for pose estimation techniques. A critical part of this process is the operation of camera calibration, which is necessary to compute the intrinsic parameters, connecting the quantities measured in the image plane to those referred to the camera frame, and the extrinsic parameters, relating the camera frame to quantities defined in a frame attached to the manipulator. This procedure characterizes the pose of the camera frame with respect to the base frame (for eye-to-end cameras) or to the end-effector frame (for eye-in-hand cameras) [26].

The 3D robotics hand-eye calibration is the task of computing the relative 3D position and orientation between the camera and the robot flange in an eye-in-hand configuration. This is the task of computing the relative rotation and translation between the two coordinate frames: one centered at the camera lens center and the other one placed on the last link of the robot manipulator [27]. Various calibration techniques exist, based on different algorithms requiring the use of calibration planes where a certain number of points can be easily detected, as can be a chessboard pattern [26].

When vision is used to measure the 3D geometric relationships between different parts of an object, it is often necessary to use the manipulator to move the vision sensor to different positions in the workspace to see different features of the object. At each point, the 3D position and orientation of the feature measured by the vision system is only relative to the vision sensor. Since the manipulator moves the sensor to different poses, the measures taken at different positions are not related among them, unless we know the 3D relative position and orientation of the sensor at different locations. If the robot system can determine where the gripper is in the robot world coordinate system, then it should track 3D motion between positions. Since the camera is rigidly connected to the gripper, it performs the same rigid body motion but only considering the robot world coordinate system. With eye-in-hand configuration, the vision system may only determine where the part is relative to the sensor, but the robot does not know how to place the manipulator to grasp it. This problem can be resolved in the robot hand-eye calibration.

Nowadays, the standard approach to calibration involves a robot equipped with a camera performing a series of motions with the camera acquiring a picture of a calibration object at the pause of each motion. The minimum number of stations is three, where station means the location where the robot pauses for doing camera extrinsic calibration. Using more than three stations improves the accuracy [27].

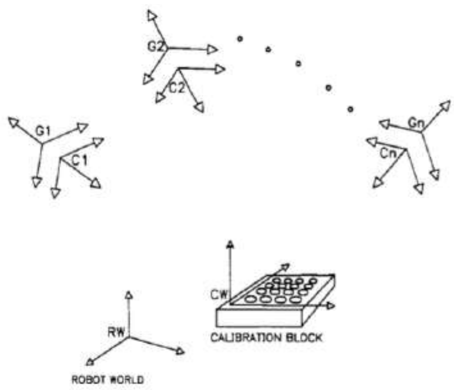


Figure 14 - Basic setup for robot hand/eye calibration. G is the coordinate frame fixed on the robot gripper, C is the coordinate frame fixed on the camera, CW is an arbitrarily world coordinate frame set on the calibration block, RW is the robot world coordinate frame [27].

The distance between the camera lens center and the calibration block has a dominant effect on the translation error. This is reasonable since the greater the distance is, the greater the effect of angular error for camera extrinsic calibration has on the position of the camera relative to the calibration block. The accuracy can be improved by minimizing the distance between camera lens center and calibration block. The error of rotation is linearly proportional to the error of orientation of each station relative to the base [27].

1.3.3 VISUAL SERVOING

It is common to talk about a robot moving to an object, but in reality, the robot is only moving to a pose at which it expects the object to be. This is a subtle but deep distinction. Consequently, the robot will fail to grasp the object if it is not at the expected pose [25].

The basic task of visual positioning is to control the pose of a robot end-effector with respect to an object observed by the camera, using information extracted from images of the robot workspace. The robot pose is a six-element vector representing the position and orientation of the end-effector in 3D space. In general, the aim is to achieve a desired pose relative the image.

Researchers have explored several different methods for computing the robot pose from feature information. Perhaps the most straightforward is 3D pose determination with 2D projection transformations together with depth estimation techniques. Depth information may also be derived from sequential views from a single camera, using techniques known as monocular stereo, motion stereo or depth from motion. Typically, robots were required to stop for a while after each movement iteration to allow the acquisition and processing of each new image. Such systems are frequently referred to as "*static look-and-move*" structures. While static look-and-move control is sufficient in some applications, real-time dynamic visual servoing is the focus of much of today's research in visual control. To avoid the additional image analysis time required for pose estimation, direct image-based feedback is often used in visual servoing systems.

An example of functioning is reported in *Figure 15*, with a simulated reference image consisting of four points. Starting from the definition a prespecified pose, relative to scene, at which a desired reference image is observed. Then, using an image-based controller, iteratively generate cartesian movement commands which bring the robot to the reference pose with respect to the image, starting from any arbitrary initial position [28].

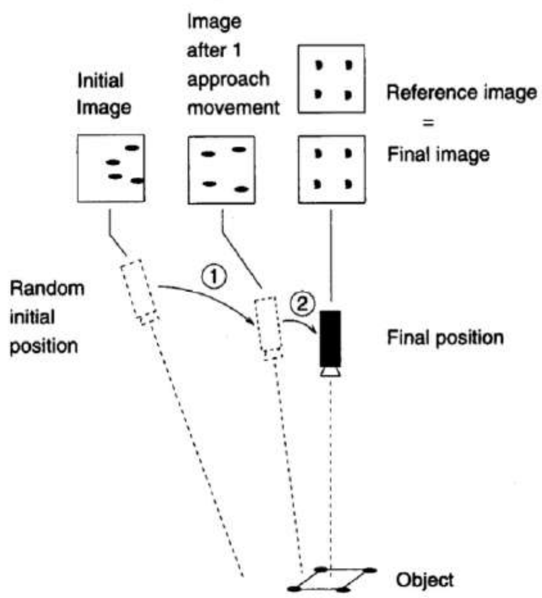


Figure 15 - Visual positioning of a eye-in-hand mounted camera using iterative approach movements [28].

Visual control has been studied extensively for industrial robot arms. Hand-mounted cameras have the advantages of proximity to the task being performed, allowing attention to avoid occlusion and increasing accuracy. This can be a drawback, since the field of view depends on the end-effector location and orientation and the focus can change as the camera approaches the workpiece [28]. A servo mechanism, is an automatic device that uses feedback of error between the desired and actual position of a mechanism to drive the device to the desired position [25].

The objective of visual servoing is to ensure that the end-effector, based on visual measurements elaborated in real time, reaches and keeps a desired pose with respect to the observed object. It is worth remarking that the direct measurements provided by the visual system are concerned with feature parameters in the image plane, while the robotic task is defined in the operational space, in terms of the relative pose of the end-effector with respect to the object.

The vision-based control schemes can be divided into two categories: those that realize visual servoing in operational space, also termed position-based visual servoing, and those that realize visual servoing in the image space, also known as image-based visual servoing. In the position-based visual servoing approach, the feedback is based on the real-time estimation of the pose of the observed object with respect to the camera using visual measurements. Its conceptual advantage regards the possibility of acting directly on operational space variables. Therefore, the control parameters can be selected according to suitable specifications imposed to the time response of the end-effector motion variables, both at steady state and during the transient [26].

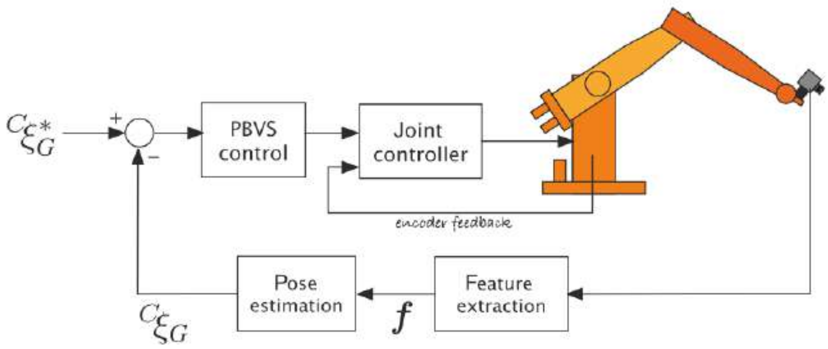


Figure 16 – Position Based Visual Servoing (PBVS) control scheme. The image of the goal is function of the relative pose ${}^{C}\xi_G$, while f represents the features extracted from the image[25].

In the image-space visual servoing approach, the control action is computed based on the error defined as the difference between the value of the image feature parameters in the desired configuration pose and the value of the parameters measured with the camera in the current pose. The conceptual advantage of this solution regards the fact that the real-time estimate of the pose of the object with respect to the camera is not required [26].

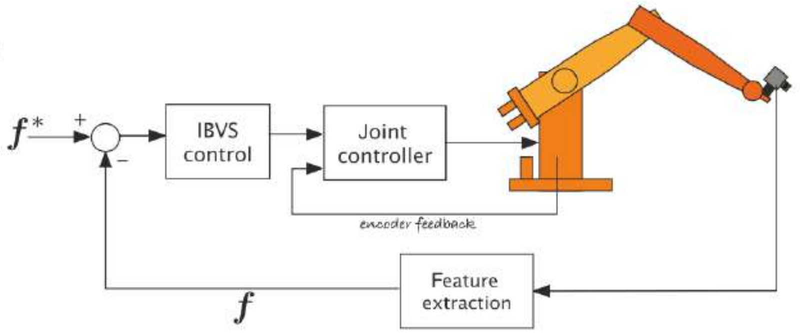


Figure 17 – Image Based Visual Servoing (IBVS) control scheme, f represents the features extracted from the image[25].

The current lack of application-driven systems is perhaps because vision-guided control still has not been proven sufficiently reliable, robust and cost-effective for many real-world problems. While cost is destined to decrease as hardware is improved, increased reliability and robustness will depend primarily on a more sophisticated use of vision [28].

CHAPTER 2 – MOBILE MANIPULATOR PROTOTYPE

The MoMa prototype discussed in this work consist of an autonomous mobile robot (AMR) equipped with a collaborative robot. This innovative setup employs a camera-based marker detection system, enabling precise alignment of the cobot movements within its designated workspace. In this section an in-depth analysis of the hardware architecture is performed, describing the main components, their integration and the communication interfaces required to enable coordinated operation between the mobile platform and the manipulator.

2.1 OASIS 600 UL MOBILE ROBOT

Standard Robots is a leading company in laser SLAM AMR R&D and manufacturing. They provide standard solution to industry, with scalable application, predictable outcome and dedicated services. Their focus is on logistics flexibility and efficiency improvement, especially for intralogistics control and material flow management among plant areas for industry leaders including TOYOTA, BOSCH, AMD, OPPO, JABIL, Red Bull.

Standard Robots

Figure 18 – Standard Robots logo.

Oasis 600UL is a standard mobile robot platform used in indoor industrial environment, with the functions of plotting environmental maps, path navigation and autonomous path planning avoiding obstacles.

The robot is based on natural trackless navigation technology, environment maps can be automatically generated without the need of scene modification, realizing rapid deployment of scheduling plan. Real-time acquisition of robot hardware and running state enables self-detection and rapid fault diagnosis.

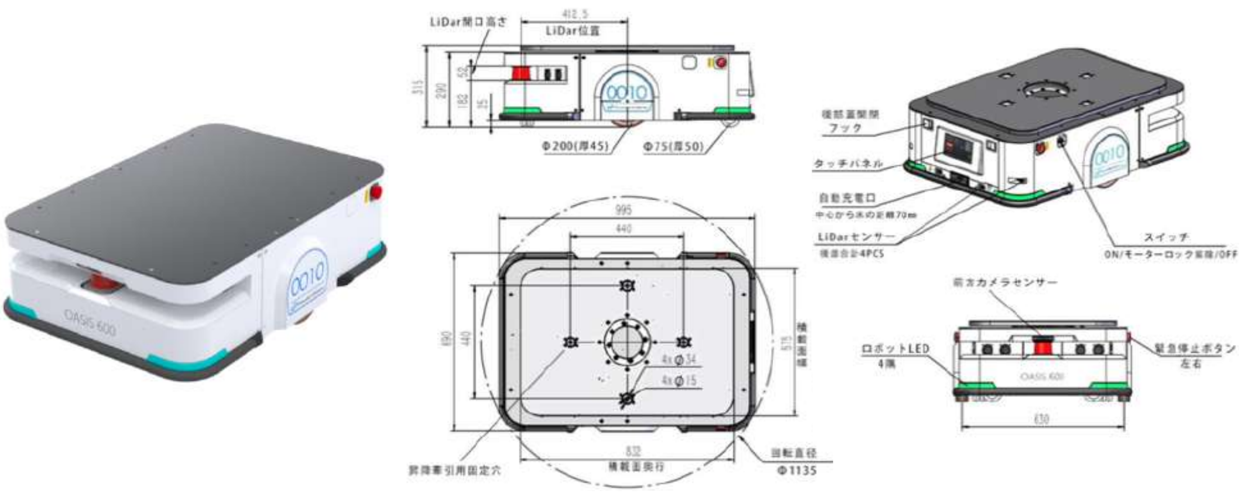
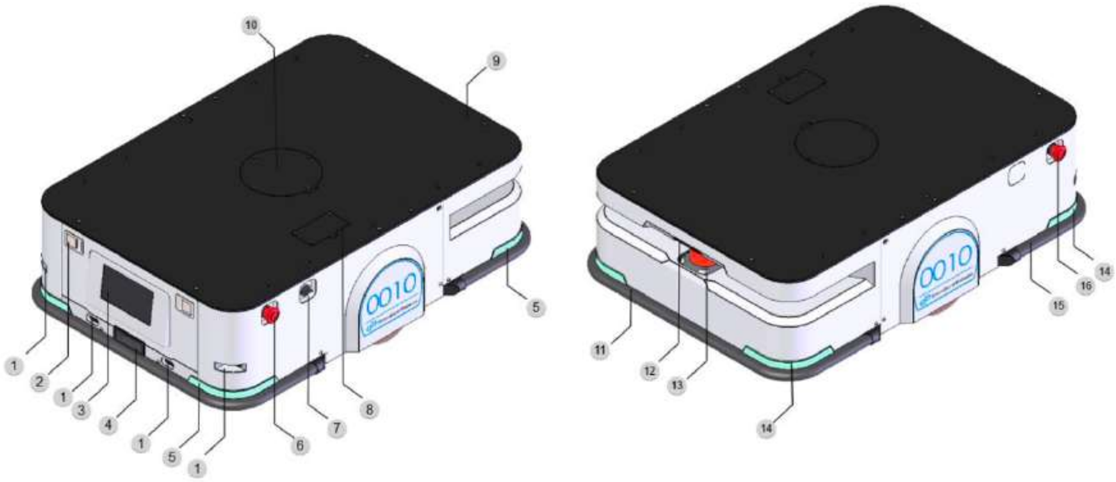


Figure 19 – On the left, the Oasis 600UL. On the right, its construction details.

2.1.1 PRODUCT SPECIFICATIONS

Oasis 600UL is made by six-wheel chassis, four universal driven wheels and two motor-driven wheels which enable the robot to move in straight and turn by the differential controller for curve directions. To realize high-precision tracking driving and smooth speed control, the robot autonomously plans the path and navigation information to control the drive wheels in real time on the map created by the user. Robot perceives environmental information within driving through the front laser radar and makes some choices. The decision-making regards safety zone switching, deceleration, stopping, active obstacle avoidance ensuring safe path and minimizing risks.

Oasis 600UL is provided with a laser radar in the forward direction of the robot that it uses to identify the surrounding environment. A 7" interactive touch screen is embedded at the back of robot. The touch screen displays robot state, user could send some work instructions to the robot to handle the general abnormalities via the touch screen. The rated load is 600kg.



1	Backward proximity sensor	2	Rear hatch cover lock	3	Touch display screen
4	Automatic charging port	5	Right indicator light	6	Right emergency stop button
7	3-gear switch	8	Closure plate of the external interface	9	Top tray
10	Middle closure plate of the top tray	11	Front safety edge	12	Visual obstacle avoidance camera
13	Laser radar	14	Left indicator light	15	Rear safety edge
16	Left emergency stop button				

Figure 20 – Detailed description of all the Oasis 600UL embedded components.

Four LED indicator lights are mounted at four side corners of the robot body, to indicate robot states. The left two and the right two are coupled, indicating via different light colors, the constant light, the blinking and the breathing. Meanwhile, one loudspeaker is equipped on the robot to synchronously make sound and light prompts in the working state. The robot body has three movable parts, including rear hatch cover part and top expansion cover plate. The rear hatch cover can be opened to charge or replace the battery. Oasis 600UL is supplied by large-capacity lithium iron phosphate batteries, enabling it to work for 8 consecutive hours. The robot is accompanied by a portable charger releasing 10A current, increasing the battery of the robot by 10%-95% in about 4 hours. As optional equipment, the automatic recharging enables robot returning to the charging station, ensuring 24/7 operation and high-frequency rapid response during tasks. The top square cover plate is an expansion interface component applicable to connect and communicate with the external devices. The Oasis 600UL standard mobile robot platform supports standard and highly interchangeable hardware expansion modules with the function extension interface to perform different types of tasks. Jobs include goods transfer through forklift module, production line transfer through conveyor module and object grasping including the arm module.

Figure 21 – Different expansion modules for Oasis 600UL. From the left, the forklift module, the conveyor module for material transfering and a robotic arm for object manipulation.

All the technical details are reported in the *Table 3*.

Type	Item	Technical indicators	Remarks
Basic performance	Overall dimensions (LxWxHmm)	995*690*290	
	Dimensions of loaded surface (LxWmm)	952*650	
	Dead weight [*1] (Kg)	<150	
	Maximum load (kg)	600	Including weight of carrier
Movement performance	Maximum speed (m/s)	1.5	
	Operational speed (m/s)	Forward: 1.34; Backward: 0.3	
	Turning radius (mm)	0	
	Gyration radius (mm)	568	
	Climbing ability	3°/5%	
	Height of obstacle avoidance (mm)	10	
	Seam crossing width (mm)	30	
	Ground clearance (mm)	25	
	Width of walking passage (mm)	Min 890	
	Width of gyration passage (mm)	Min 1300	
	Station positioning accuracy (mm)	±10	
	Precise connection accuracy (mm)	±5	
Endurance performance	Battery capacity	51.2V40Ah	Lithium iron phosphate battery
	Battery service life (times)	DOD≥80% 1500	0.5C charging 1C discharging (normal temperature)
	Endurance (h)	8	1m/s, 600kg load
	Charging mode	Automatic + manual	Manual: maximum current 10A Automatic: maximum current 30A
	Charging time [*3] (h)	1.3	Battery bar from 10% to 99%
External interface [*3]	Power output port	2 circuits of DC51.2V (40~57.6) 2 circuits of DC24V (regulated power supply)	
	Standard communication interface	1 circuit of RS232 1 circuit of CAN	
	I/O interface	Support CAN communication expansion.	

*1 The dead load of the robot may differ on selected accessory modules;
*2 The charging time is measured by using a 30A charging pile at an ambient temperature of 20°C;
*3 Please contact Standard Robots to get User Manual for detailed external interface usage.

Table 3 – Technical details of the Oasis 600UL.

2.1.2 POSITIONING AND NAVIGATION

The simultaneous localization and mapping (SLAM) technology is used for Oasis 600UL mobile robot. Since the 2D laser radar can only scan at the specific height, it cannot detect objects higher or lower than this plane. Laser radar detection radius ranges from 0.1 m to 30 m with detection angle within 240°. Different reflection effects may produce great interference to the specific detection for objects made of special colors and materials, which can cause the positioning deviation of ±10 mm.

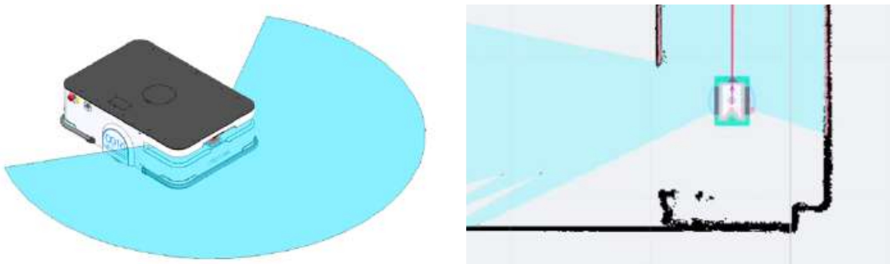


Figure 22 – On the left, a representation of the detection angle of the laser radar. On the left, the visualization of the scanned area during the operations.

Good performances are guaranteed for different environments. In static scenario, the floor equipment and articles are not moved and there is no area for free activities of personnel. The main moving objects in the scene are vehicles and robots. In highly dynamic areas, articles are temporary stored and may move at any time, which could lead to change in the environment, posing risks of positioning error and getting lost because of frequent human activities.

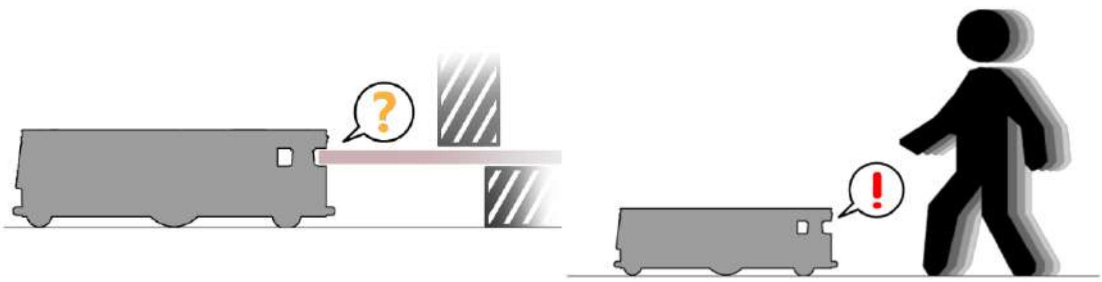


Figure 23 – On the left, the plane in which the robot can detect obstacles. On the right, the robot in presence of humans in a dynamic environment.

2.1.3 OBSTACLE AVOIDANCE

Oasis 600UL is configured with various obstacle avoidance sensors, which allow the robot to identify potential risks and activate procedures to slow down or stop its movement to avoid collision with people or objects, ensuring safe work. The robot is equipped with the following sensors: laser radar, rear proximity sensor detection set and vision-based obstacle avoidance module.

The 2D laser radar installed on the front groove has three main functions: mapping the working area, providing the robot location data in the environment and continuously analyzing the environmental information when the robot is working to avoid collisions with people or objects. The laser radar has a 240° angle view, it cannot detect objects beyond this angle. The maximum distance that it can perceive is 30 m, detecting objects about 200 mm above the ground.

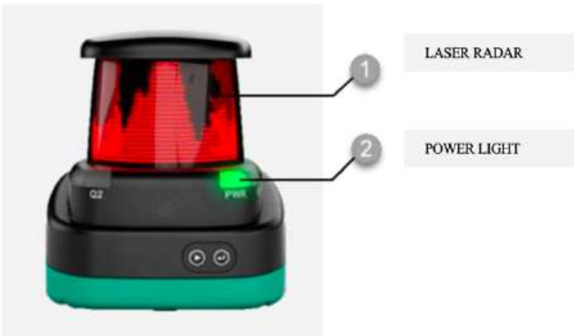


Figure 24 – Laser radar mounted on the front groove of the Oasis 600UL.

The rear proximity sensor detection set is installed in the back of the robot, which enables the robot to detect obstacles when moving backwards, with a 1.8° cone detection angle and ranges from 100 to 1500 mm. The detection height of the two proximity sensors in the middle is 70 mm, while for outwards is 130 mm.

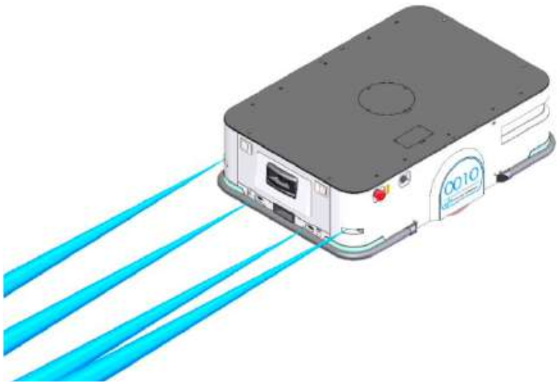


Figure 25 – Proximity sensors installed on the rear grove of the Oasis 600UL.

The vision-based obstacle avoidance module is installed in the front of the robot. It is mainly used to identify the objects which are at low heights and in front of the robot, enhancing the obstacle detection ability of the robot. It has a 90° scan view; the size of the smaller object that can be detected by system is 35 mm (H) × 20 mm (W). The data acquired by this module within 150-1500 mm are used as the obstacle avoidance information, looking for objects about 30 to 245 mm above the ground.

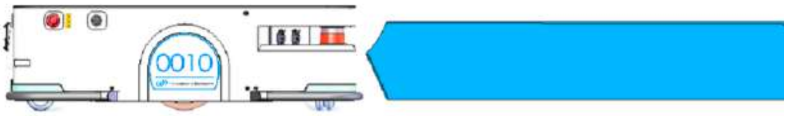


Figure 26 – Vision-based obstacle avoidance module installed on the Oasis 600UL.

When robot detects obstacles in the movement path, it will switch between two system states: slow down when the obstacle is in the deceleration area or suspend when the obstacle is in the stop area. In the first case, state indicator light blink in yellow, in the second the state indicator light changes in constant light. Robot has two obstacle avoidance strategies after entering the state of "*Obstacles detected, suspend*": the default strategy is "*Suspend and wait*", the robot suspends and waits for the obstacles to be removed when encountering obstacles during its movement. Another strategy is "*Replan the path*", the robot will replan the path when encountering the obstacles to be removed. Obstacle avoidance is only activated in the direction of movement: rear detection is disabled when moving forward and front detection is turned off when moving backward.

The robot creates a detection area to identify obstacles in real-time based on its planned movement path. The *Figure 27* shows how the linear movement obstacle avoidance areas are spatially located with respect to the robot. Red area is a stop area, the robot will trigger deceleration to stop, while yellow area is a deceleration area where the robot will trigger deceleration to the set allowable constant speed of 0.4 m/s^2.

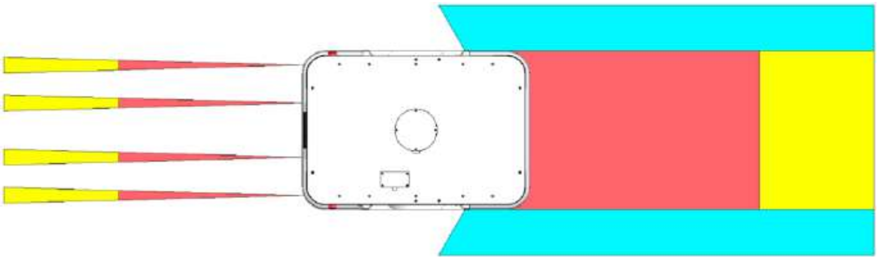


Figure 27 – Representation of spatial location of obstacle avoidance areas relative to the robot.

2.1.4 SAFETY

User must tap the emergency stop button to trigger emergency stop in hazardous situations. One button is respectively provided at left and right sides of the robot. The front and back of the robot are provided with safety edges. When the safety bumpers collide with the surrounding objects, they are squeezed and then the emergency stop state is triggered. In case of emergency button triggered, the robot executes Type 0 emergency stop, aborting the movement and turning off the power. The indicator lights around the robot become constantly red. User must manually turn the reset emergency stop button and then the robot will switch to the "*emergency stop recoverable*" state. User must release the emergency stop state by clicking the reset button on the on-board interactive touch screen or by releasing the emergency stop button in the software view.

2.2 JAKA ZU 5 COLLABORATIVE ROBOT

JAKA Robotics is established as a spin-off of the Robotics Institute of Shanghai Jiao Tong University since 2014. In recent years, JAKA becomes a global robot company with over 200 overseas customers. The global expansion kick-off with investments in overseas markets, opening the European Headquarters in Germany in 2022.

Figure 28 – JAKA Robotics logo.

This company ensures easy implementation of their manipulators. There is no need to know any professional programming language or any prior programming experience, users simply need to learn the intuitive graphic programming software. With wireless connection it is easy to communicate and assign tasks to a cobot, allowing anyone to set and adjust positions. Users can connect the robot creatively using a smart mobile terminal paired with the JAKA APP, enabling a single mobile terminal to control multiple robots and eliminating the need for traditional teaching pendants. The APP is compatible with tablets, smartphones and PCs. The user only needs to manually guide the robot to complete the programming, which greatly improves production efficiency. These cobots perform several applications of all industries thanks to their flexibility. They meet the needs of reliable and efficient operations in high-precision collaboration scenarios.

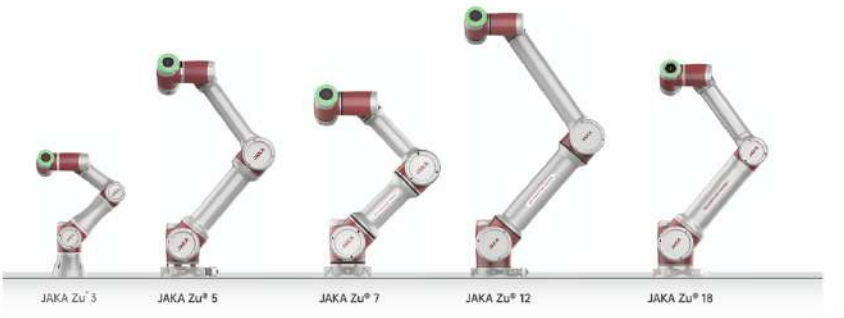


Figure 29 – All the products of the JAKA Zu series.

2.2.1 PRODUCT SPECIFICATIONS

JAKA Zu 5 is an intelligent, light weight, modularized collaborative robot with a payload of 5kg. It is an industrial collaborative robot suitable for use in industrial environments. The robot consists mainly of six joints and two aluminum tube arms.

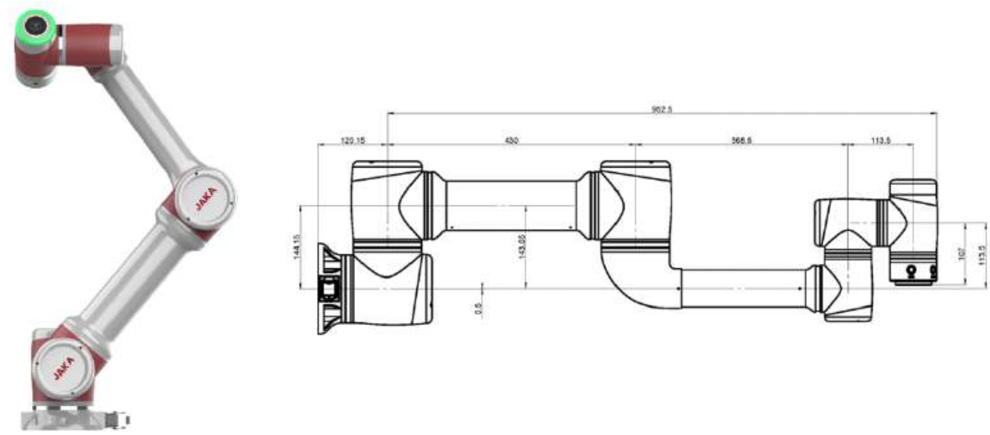


Figure 30 - On the left, the JAKA Zu 5. On the right, its construction details.

The base is used to install the robot through four M8 bolts. It is necessary to fix the robot on a sturdy surface that shall be strong enough and vibration free. It can be mounted in various ways, such as ground, wall, and celling mounting. If the robot is mounted on a moving platform, then the acceleration of the moving mounting base shall be very low. A high acceleration might cause the robot to stop, thinking it bumped into something. When choosing the robot installation position, the cylinder space directly above and below the robot base must be considered. This zone is critical since when the tool moves too close to it, the robot joints are forced to move very quickly causing the robot to work inefficiently and making it difficult to conduct a risk assessment. A well-performing behavior is obtained by moving the robot across the entire workspace.

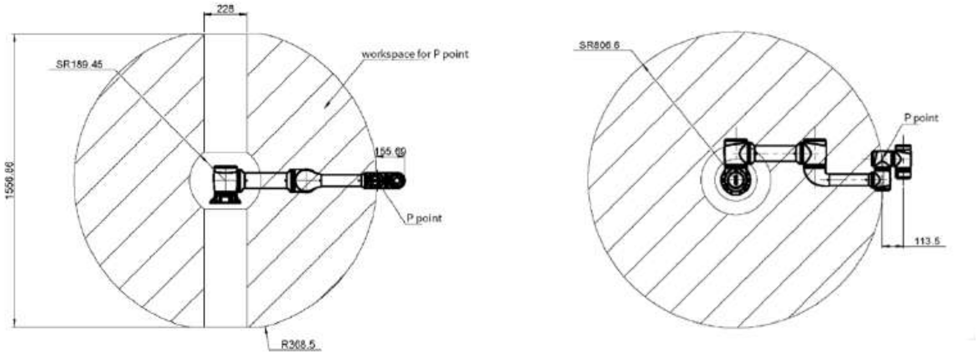


Figure 31 - Technical details of the robot workspace.

The flange end is used to mount the tool, which can perform translational and rotational movements in the robot's working range. On the MoMa prototype, a simple industrial tool is mounted, with one gripper on the bottom, in the same direction of the camera and one on the right side.

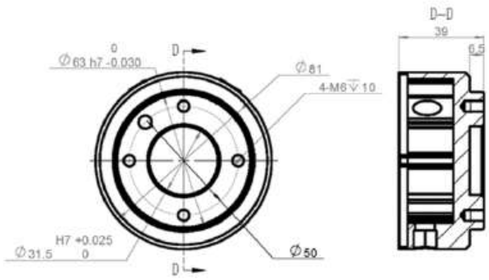


Figure 32 - Flange end construction details.

All the technical details are reported in *Table 4*.

	Product parameters	JAKA Zu® 5	
Product features	Maximum payload	5kg	
	Weight	23kg	
	Working radius	954mm	
	Repeatability	±0.02mm	
	Number of axis	6	
	Programming	Drag teaching and graphic programming	
	Teaching pendant	PC, mobile (PAD/mobile)	
Working range and speed	Robot joint	Working range	Maximum speed
	Joint 1	±360°	180°/s
	Joint 2	-85°,+265°	180°/s
	Joint 3	±175°	180°/s
	Joint 4	-85°,+265°	180°/s
	Joint 5	±360°	180°/s
	Joint 6	±360°	180°/s
	Maximum speed of the tool end	/	3m/s
Specifications	Power consumption	350W	
	IP classification	IP54	
	Tool I/O ports	Digital input 2	
		Digital output 2	
		Analog input 1	
	Base diameter	158mm	
Electrical cabinet	IP classification	IP44	
	I/O ports	16 digital inputs, 16 digital outputs, 2 analog inputs or output	
	Communication	TCP/IP, Modbus TCP, Modbus RTU, Profinet, Ethernet/IP	
	Power	100-240VAC, 50-60Hz	
	Size	410×307×235 (mm) (W×H×D)	
	Weight	15.4kg	

Table 4 - Technical details of the JAKA Zu 5.

2.2.2 EQUIPMENTS AND SAFETY

Behind the fifth joint, at the end of the arm, a ring indicator and a button used for TIO interface are located. They can display different colors, each indicating to a specific status of the robot. When the robot is running program, press the pause button could pause and resume the program.

On the side of flange located two more buttons: FREE button and POINT button. When the first one is pressed, the robot would enter free-drive mode. The second one could be used with the robot APP: when this button is pressed the robot position would be recorded in the program.

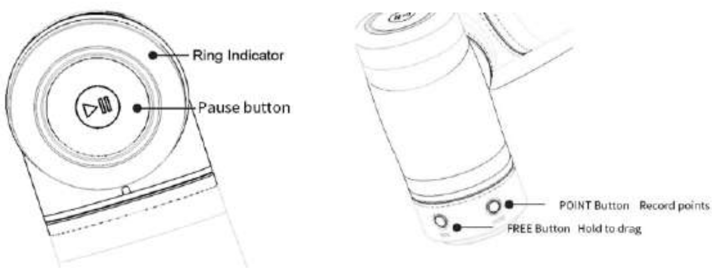


Figure 33 - Graphical description of the available control buttons on the robot.

The direct control on the robot can be assumed by an external stick: when the programming job is finished, the stick could be sued to govern the robot. Commands can be sent to the robot using buttons on the stick.

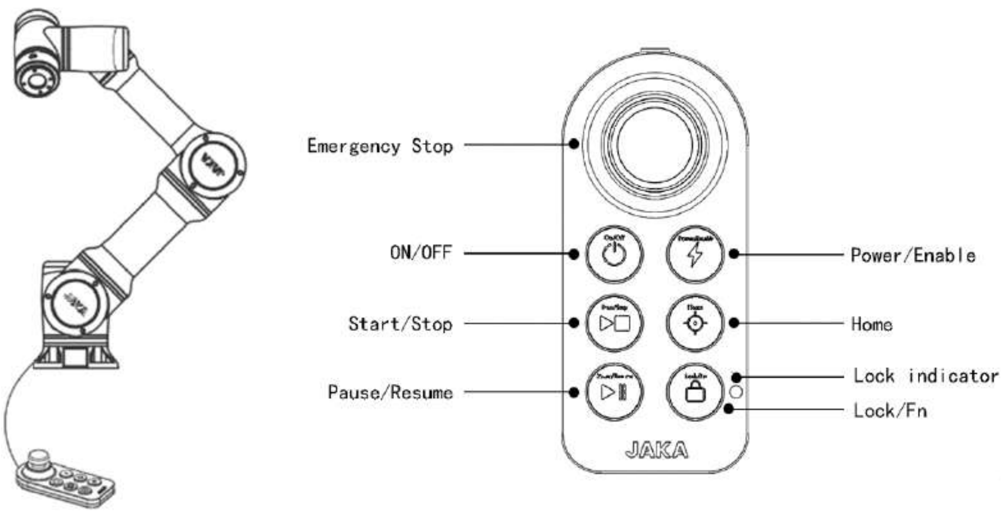


Figure 34 - On the left, the control stick connection. On the right, the description of the available functionalities.

The robot is equipped with special safety-related features, which are purposely designed for collaborative operations, where the robot operates without fences and together with a human. JAKA cobots are designed to work safely with humans thanks to collision detection, enabled by a built-in torque feedback module to avoid harm. When an emergency occurs, the pressure of the emergency stop button will stop all movement of the robot immediately. Emergency stop cannot be used as a risk reduction measure, but as a secondary protective device. In the unlikely event of an emergency where robot power is either not possible or unwanted, the robot joint can be forced to move through manual brake release. Remove the joint cover by removing the screws that fix it. Press the plunger on the small electromagnet to release the brake, then move the robot arm manually paying attention to gravitational pull which can cause the arm to fall.

2.3 JAKA LENS 2D CAMERA

JAKA Lens 2D integrated camera adopts 2D high-resolution industrial camera. It features a professional design, small and lightweight, showing a delicate appearance. It can realize 2D vision function by the external fixed installation or by installing it at the end of the robot.

Figure 35 - JAKA Lens 2D.

The crucial step for camera integration into the system is a secure and stable installation of the device on the cobot end flange, by using the proper adapter. The most common set up ensures that the camera is aligned along the Z-axis direction of the terminal flange of the robot. If the installation cannot be completed under this condition due to working restrictions, a calibration step is required. In the MoMa prototype, the camera focuses points on Y-axis direction of the flange. This assembly design is due to the presence of a large gripper, where the vision system with its support is mounted. After the camera is assembled, fix it at the end of the robot in a way as compliant as possible to guarantee accuracy during the usage.

Figure 36 - On the left, the mounting layout suggested from the manufacturer.
On the right, the actual position of the camera on the robot gripper.

The camera cable is a composite cable (including network cables and power cords). Connect the network cable at the end of the composite cable to a gigabit router/switch on the same network as the robot. The other two power cables are connected to 24V in white and 0V in black. When the camera power supply is normal, the indicator light is green. During the installation, it is recommended that the wiring harness interface does not obstruct the arm motion and helps facilitate the cable organization.

Figure 37 - Graphical instructions for cable connection.

The camera is equipped with a special light source module. It is also possible to adjust the focal length of the camera, you first need to set up the photographing position of the robot to ensure that the height and the visual field have met the photographing conditions and then proceed by unscrewing the camera cover and manually set up the desired settings for aperture and focal length.

Figure 38 - Graphical instructions for manual settings.

This vision system can be used to perform image acquisition, template matching, code scanning, edge and circle identification, line intersections, distance calculation, monocular measurement, blob extraction, character identification and color recognition.

The camera overall parameters are reported in *Table 5*.

Lens	Lens 2D CGC500-F08
Resolution	2592×1944
Max frame rate	24fps
Data interface	Gige
Color mode	Black and white / color
Lens focal length	8mm
Object distance	>100mm
Vision	>70×50mm
Precision	>0.08mm
Image processing	Soft-trigger image acquisition, single frame processing time within 1s

Table 5 – Technical details of the JAKA Lens.

2.4 SYSTEM INTEGRATION

System integration in a MoMA involves seamlessly combining multiple subsystems as a mobile base (AMR), a collaborative robotic arm and often advanced vision and sensor systems, with the final goal of working together reliably and safely in a cohesive system. Effective system integration requires tight synchronization between the AMR and the manipulator. This is achieved through robust communication protocols and advanced control algorithms that coordinate actions performed by the two robots. Successful integration depends on the harmonious interfacing of hardware (motors, sensors, batteries) and control software. For the MoMa examined in this research, a modular design approach has been noticed, since every component is tuned to work independently.

The first operation was to remove the covers of the integrative structure to better analyze the wiring. Once all the connections were identified, it was possible to define an electrical scheme, useful for representing the communication between components.

The brain of the entire system can be identified in the JAKA MiniCab. It plays a key role in controlling the robotic arm and managing I/O signal exchange, configuring communication across different modules. This component directly controls the valve blower and the bottom gripper. An extension I/O module is connected to the cabinet through Modbus communication. The *Figure 39* represents a schematic of the architecture of the system integration.

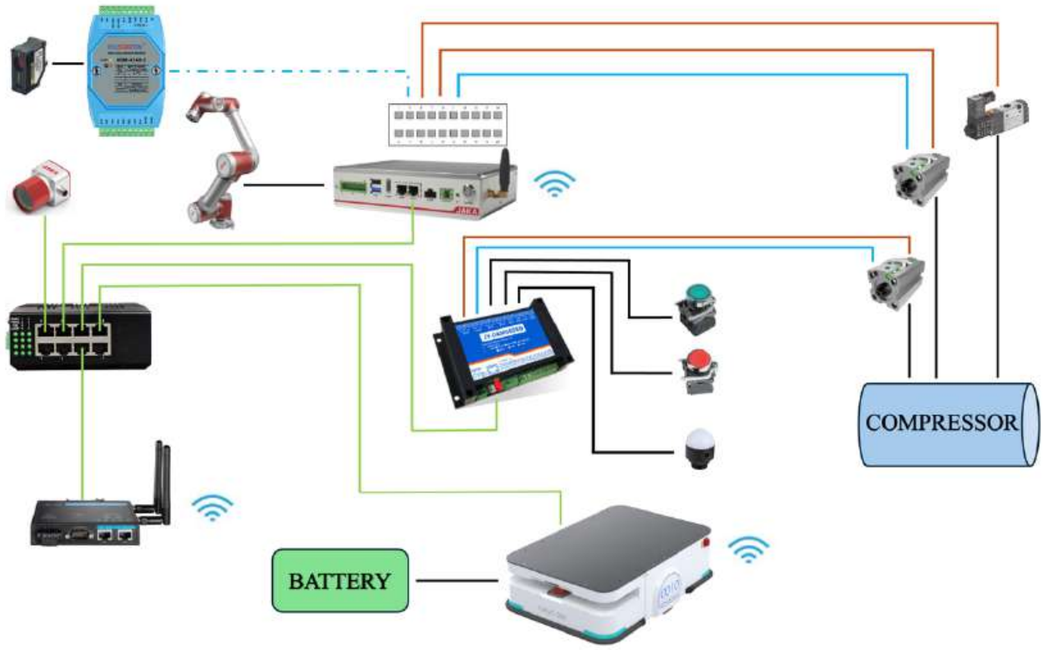


Figure 39 - Schematic representation of internal wiring and connections.

The internal layout presents a structural division between the electrical and pneumatic parts. The entire MoMa is powered by the AMR battery, since no other wiring connections are possible. As one might expect, the power autonomy significantly decreases due to the consumption of all the electrical modules. A mobile robot that autonomously returns to the charging station during idle periods may increase the efficiency of the entire system. This ensures that the MoMa battery remains charged, preventing interruptions during task execution that could potentially interfere with its performances.

Focusing on the pneumatic system, a small compressor is onboard, equipped with its own motor for autonomous recharging even during task execution, eliminating the need to stop for refill or human intervention. This system is used to feed the dual pneumatic gripper controlled by two electro valves and the air blow which is suddenly used to clean parts before grasping.

2.4.1 JAKA MINICAB CONTROLLER

JAKA MiniCab is a compact control cabinet with wide-voltage DC power supply, which can be used with JAKA Zu robots. It can seamlessly integrate in the system, providing powerful performance for numerous applications. MiniCab has user interfaces on the front panel and side panel. The front panel includes I/O, two USB, a HDMI, two LAN, a control stick and an E-STOP interface, a Wi-Fi status indicator and a POWER button. The side panel includes a power input interface, a robot connection cable interface, a 2.4G Wi-Fi antenna, and a network reset button. The front panel of the control cabinet that has a 20 PIN I/O interface for external connections also provides safety function input interfaces, where the emergency stop function can be enabled by a proper configuration. Users can connect safety doors, safety light curtains and sensors according to requirements.

Figure 40 - JAKA MiniCab controller.

The cabinet needs to be installed in a dry place with good ventilation and cooled by natural convection. In situations where significant heat is generated, such as when the robot operates at high speeds, carries heavy loads, or brakes frequently, an external fan should be used to cool down the MiniCab. There are three mounting methods available: bracket mounting, bottom mounting or guide rail mounting as in MoMa considered in this work.

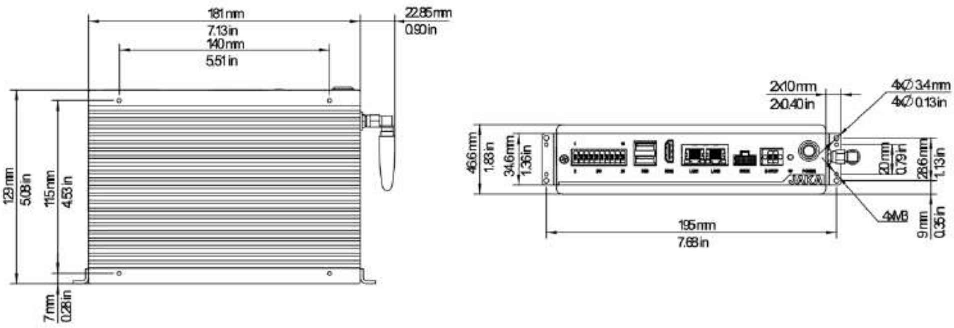


Figure 41 - JAKA MiniCab construction details.

The control cabinet is equipped with a Wi-Fi module that allows the operation terminal to connect with the control cabinet to control the robot. It is also accessible to connect the network port of the control cabinet to the router and the operation terminal to the wireless network of this router at the same time. Due to its compactness and connection facilities, it makes the integration convenient and simple. The JAKA MiniCab control cabinet can be used with mobile platforms such as AGV and can be powered by the 48V power battery inside the AGV.

Digital I/O has 7 digital signal interfaces, each I/O channel simultaneously has NPN input and output functions. PIN17 and PIN18 on I/O interface are RS485 communication interfaces, which can be connected to external devices for communication.

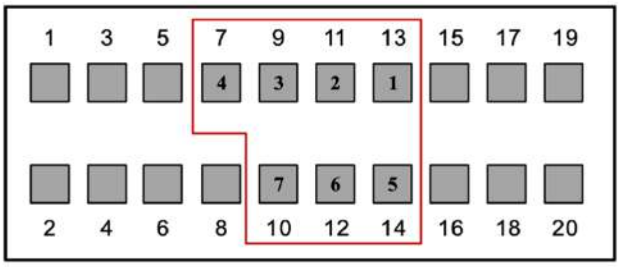


Figure 42 - JAKA MIniCab I/O interface.

The user can set the functioning of each interface in the APP, ensuring that only one interface type (DI/DO) can be selected for each channel at the same time. For the prototype in analysis, the wiring and respective functions are configured as follows.

Controller IO assignment
DO1 -
DO2 - Bottom gripper ON close /OFF open
DO3 - Valve Blower
DO4 -
DI5 - Bottom gripper CLOSE
DI6 - Bottom gripper OPEN
DI7 -

Figure 43 - Input/Output assignment.

The RS485 communication is connected with the ADM-4240-C external device. This module is responsible of converting the input voltage received from the cabinet to appropriate levels to power a laser pointer, ensuring that the correct voltage is supplied. The laser pointer is attached close to the camera, it serves to verify the accuracy of the manipulator movements in during testing procedures.

The GND pin of the DC input connector and the IO terminal are internally connected via 0 Ohm resistors, allowing a maximum current of 10A. Ensuring the same voltage level between these GND points is critical. To protect the MiniCab, an additional shunt cable (diameter >1 mm²) must be connected between the GND pins and the voltage between points A and C must read 0V before closing the loop. Moreover, if the AMR chassis is improperly isolated and connected to battery V+, the MiniCab may be damaged. To prevent this, it is recommended to use an isolated DC-DC power supply, as shown in the diagram. Since AMR battery voltage (48V nominal) may rise during charging, this system also ensures the MiniCab voltage remains below 58V to avoid overvoltage errors. This component is not installed on the prototype.

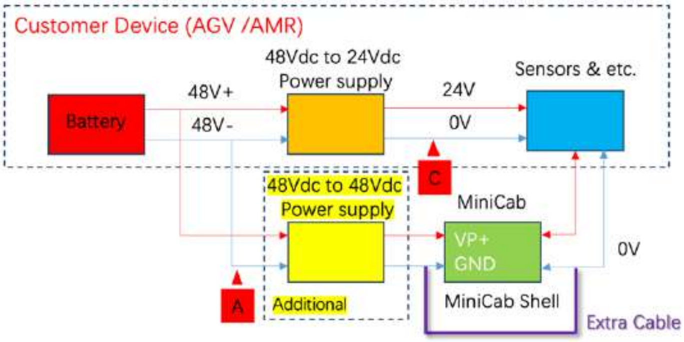


Figure 44 - Electrical scheme.

The detailed technical information of the JAKA MiniCab are reported in *Table 6*.

Model	MiniCab
Weight	1.1 kg (2.43 lb)
Dimension (W×H×D)	180×46.6×128 mm (7.09×1.84×5.04 in)
Temperature	0-50°C (32-122°F)
IP classification	IP20
Material	Steel, aluminum alloy
Power supply	DC 30V~60V
Communication mode	TCP/IP, Modbus TCP, Modbus RTU, PROFINET (1.7 App), Ethernet/IP (1.7 App)
I/O port	7 channels I/O multiplexing
I/O power supply	DC 24V

Table 6 - Technical details of JAKA MiniCab.

2.4.2 JY-DAM0808D I/O EXPANSION MODULE

The JY-DAM0808D is a digital input and output expansion, allowing additional I/O channels to control external devices. It provides 8 digital inputs (DI) and 8 digital outputs (DO), extending the capability of the MiniCab to interact with the surrounding environment, handling several peripherals required for automation tasks performed by the MoMa. This module is powered through a 24V DC supply, typically provided directly from the cabinet. The controller communicates with the module through Modbus TCP protocol, ensuring reliable data exchange. In every industrial application, this component is essential to send signals and acquire feedback from embedded sensors, increasing process reliability, flexibility and safety.

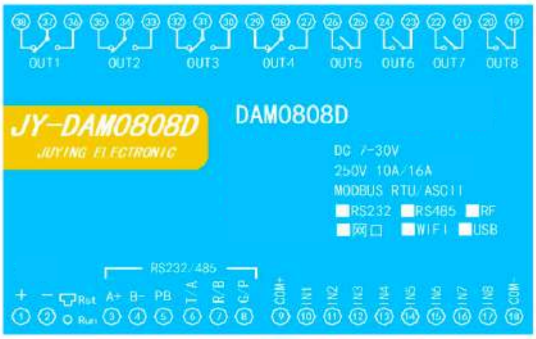


Figure 45 - Juying JY-DAM0808D I/O expansion module.

The configuration of the I/O signals is managed and monitored directly through the JAKA APP. From a functional perspective, the digital outputs are used to activate peripheral components, such as pneumatic valves, relays and signaling devices. The logical signals are associated to external devices as reported below.

Modbus extension IO assignment (DI & DO both start from 0, quantity x 8)	
DI6 - 1st button GREEN	DO4 - 2nd button indicator RED
DI7 - 2nd button RED	DO5 - 1st button indicator GREEN
DI8 -	DO6 - Three-color indicator RED
DI9 -	DO7 - Three-color indicator YELLOW
DI10	DO8 - Three-color indicator GREEN
DI11	DO9 -
DI12	DO10 -
DI13	DO11 - Side gripper ON close/OFF open

Figure 46 - Input/Output assignment for modbus expansion module.

2.4.3 ETHERNET COMMUNICATION

In the system, a switch unit allows the communication among camera, AMR, control cabinet and extension module. It ensures efficient data transfer and network stability amid these components, achieving seamless coordination and control. The switch supports high-speed Ethernet connections, which are essential for handling the real-time data requirements of the mobile manipulator. Ethernet communication provides a reliable and scalable solution, offering low-latency and high-bandwidth connections that are crucial for tasks like real-time images feeds from the camera and managing I/O signals.

192.168.71.102	Modbus extension IO module, Modbus TCP port 10000
192.168.71.26	Robot controller
192.168.71.50	Standard Robot AMR, Modbus TCP port 502
192.168.71.31	JAKA Lens 2D

Figure 47 - Ethernet IP assignment.

The TP LINK TL-CPE1300D wireless router acts as an external network access point. It is designed for industrial-grade reliability, operating across a wide temperature range and with resistance to electromagnetic interference, making it ideal for any environments. It can work in Client and Client-Router modes, with high transmission power, using dual-band enhanced roaming technology to maintain a stable and reliable communication even for mobile devices. In the examined MoMa, the gateway is not configured for external network access.

Figure 48 - System communication network.

CHAPTER 3 – SYSTEM SET UP

A mobile manipulator composed of a cobot and a mobile platform has much larger workspace than a fixed-base manipulator, due to the mobility provided by the AMR. A central issue in the development of mobile manipulator system is vehicle/arm coordination. While the on-board manipulator performs manipulation tasks, the role of the mobile platform is to position the manipulator in a precise operating point. However, to fully utilize the advantages offered by the MoMa, it is necessary to understand how properly and effectively coordinate the motions of the mobile platform and the manipulator. Between these two robotic systems there is a dynamic and hierarchical interaction, the vehicle/arm system can be viewed as mechanism resulting from serial combination of two sub-systems programmed independently to achieve synchronized tasks. In this section, the software modules are described, introducing also the communication protocols between them. Finally, the MoMa set up phase is described, with the control flow necessary to orchestrate the vision-based alignment with the workspace.

3.1 MATRIX ENVIRONMENT

Matrix is an interactive interface between users and robots, which is built into the robot main controller and is compatible with multiple types of terminals such as PCs, tablet or mobile phones, through which users can debug the robot, edit the road network, and create tasks.

Figure 49 - Description of the Matrix environment.

Oasis 600UL wireless network communication module can be set to the wireless access point mode, the default connection IP address is 192.168.71.50. Connect the terminal to the robot WIFI access, open the browser, input robot IP address in the browser address bar and execute the log in with username and password.

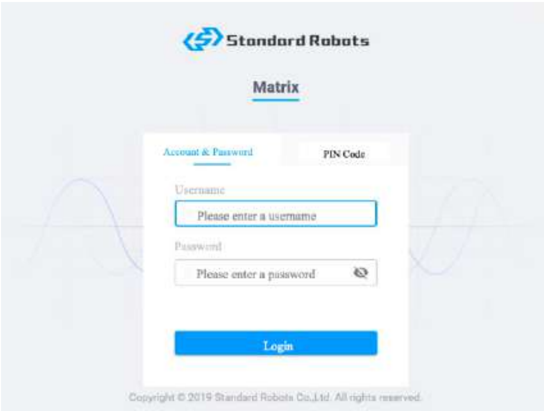


Figure 50 - Matrix login window.

After logging in, the Matrix shows its main interface and main areas: in the middle there is the map display area, with debug buttons and state bar on the top. On the left side there is the main menu bar, while on the right side the task queue and system state are indicated.

3.1.1 MAP CREATION

The map represents a 2D model of the scene where the AMR will move. The accuracy during the creation of a new map is crucial to achieve precise positioning of the mobile robot. This procedure must be executed every time the environment changed, to better address the new positions of structures in the space.

As soon as the "*Create a new map*" button is clicked, a floating directional control key appears on the page and the speed will be automatically limited to 0.5 m/s. During the map drawing, the user manually controls the robot to move in the working space, the laser radar installed in the front of the robot continuously scans the operative area, generating the 2D plane map representation. In manual control mode, the active obstacle avoidance function is disabled, so the user should observe the operating environment to avoid collisions with surrounding people or objects. To increase the accuracy in slightly complex environments, the map should be drawn moving the robot toward space features and rotate of 360° close to features such as corners, walls and cluster areas. Once the scanning operation is completed, control the robot to return in the vicinity of the starting position and ensure that the orientation of the robot is roughly the same as its initial orientation. Finally, the map can be saved.

The main purpose of the map is to draw road and stop stations for the robot. Due to the dynamic working environment, the drawn can be modified to erase obstacles such as movable obstructions that can be removed. The values of acceleration/decelerations along the path can be determined, also defining prohibited areas to better adapt robot missions to the industrial framework. Some environment information, such as walls, doors or posts can be marked on the map. Before proceeding, it is necessary to locate the robot, selecting its position and orientation on the map; then, wait for the robot to be successfully located. During the motion, the robot continuously scans the surrounding environment using a laser radar. By comparing the real-time environmental profile with the map stored in its master controller, the robot calculates its current position and orientation within the working area. The movement along a predefined path can be realized to ensure controlled and safe operation by establishing the road network with the introduction of straight or Bézier curve lines. Eventually, stations can be introduced in the map by directly interacting with the environment or by accurately defining the position coordinates and orientation. The set of the available editing tool is reported in *Figure 51*.

Figure 51 - Map editing tools.

3.1.2 TASK CREATION

The mobile robot permits the definition of a task, for the applications in which it is used independently. Entering the task editing page, it is possible to generate a mission as an execution queue that should be added between the "*Start*" and the "*End*" blocks. Then, drag and drop the hollow circle of the previous step to the following one to complete the step connection. The tasks are bound to the respective map, mainly oriented to the motion execution or signal exchanging.

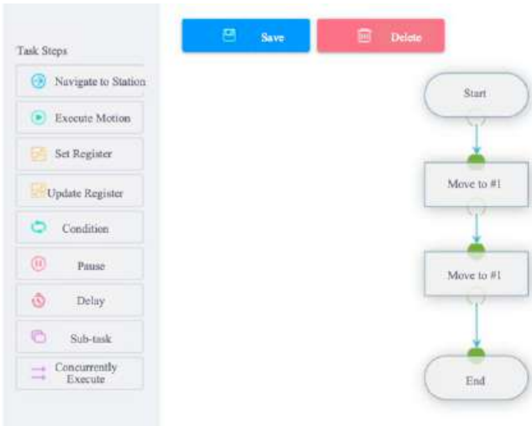


Figure 52 - Task creation interface.

3.1.3 MODBUS TCP COMMUNICATION

Since the interaction between the AMR and control cabinet is normally via Modbus TCP for motion control, an initial test has been useful to understand how this communication protocol works.

Figure 53 - Modbus logo.

This protocol was created in 1979 by the *Modicon* company for communication within PLCs. Born to be used in the industrial sector, over time this protocol has also been used in other sectors, becoming one of the most widespread. Even today, despite being more than 40 years old, it is still adopted in many devices.

Modbus TCP is an ethernet-based protocol used to exchange data between electronic devices. It offers higher data transmission speed compared to traditional Modbus. The communication occurs in a client-server mode, where a client sends a request to the server, which responds with the requested data. This protocol is widely used in industrial automation due to its simplicity, ease of integration and support across a wide range of devices. Modbus TCP is particularly advantageous for applications requiring fast and reliable communication, such as process control or energy management systems.

The structure of the message is made by a function code which defines the action to be performed and data that must be requested or written. For the TCP variant it is not necessary to specify the address of the slave, since it is directly managed by the protocol itself. These data are stored in registers, their type determines how they are accessed. From the datasheet, it is possible to observe how the constructor created the Modbus interfaces. Registers are defined as follows:

- Coil registers (00001) are binary values representing discrete outputs states which can be read or written. They can be used to pause, resume or stop the navigation;
- Discrete input registers (10001) are binary values representing discrete input states which can only be written. They can be used to observe if the robot is charging or if the emergency stop is triggered;
- Input registers (30001) are 16-bit values representing analog input data which can only be written. They can be used to capture data from the system as the temperature or the battery charge percentage;
- Holding registers (40000) are 16-bit values representing both analog input/output. They can be used to manually define the position coordinates where to send the robot, its velocity and acceleration.

For testing purposes, data values can be assigned by connecting an external tool, which simulates the Modbus TCP client, to the robot IP address. This test is performed to assess the client-server communication. Initially, the map configuration of the operative area is required. Then, three stations can be defined to assign specific robot missions, with each station identified by a unique number. A simple task is defined in the Matrix interface, where the motion directed to a specified station depends on the value assigned to the common register 40033. Since the communication worked properly, the test has been completed successfully.

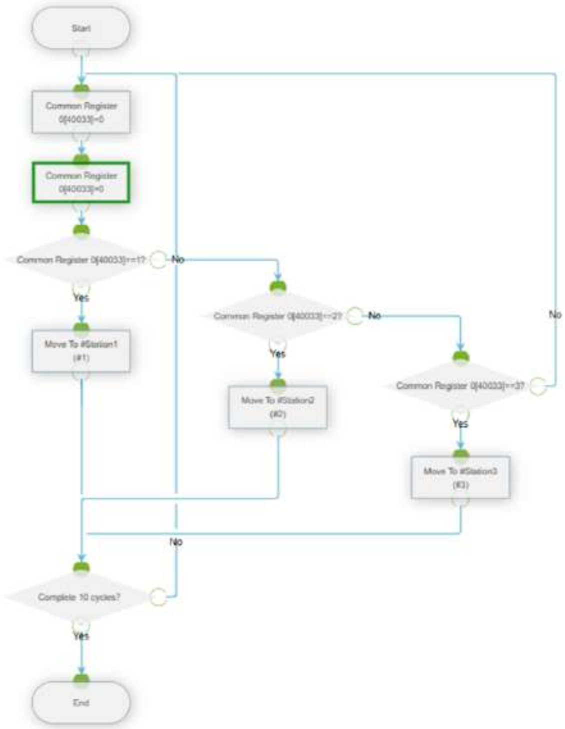


Figure 54 - Task creation for Modbus connection test.

At this point, referring to the datasheet, the register 40015 has been identified as the one responsible for robot autonomous navigation to station. The new test consists in writing the desired number of the station towards which the robot must proceed in this register. As a result, the AMR moves to the respective station configured into the register through the client tool. Even the execution of this test has been satisfactory.

3.2 JAKA APP

The JAKA APP is an APP with integrated robot demonstrator functions. The software allows the manual operation, programming, parameter configuration, information monitoring of the robots and I/O modules handling. After the installation of the software on Android or IOS mobile devices or Windows PCs, the connection to the control cabinet Wi-Fi network is required. From the main page of the APP, it is possible to select the robot to connect, either directly from the "*Online Connection*" list or by manually entering the robot's IP address in the "*Offline Connection*" window. Then, the access through username and password is necessary.

The home of the JAKA App is composed of:

1. Menu bar

2. Switch bar

3. Function bar

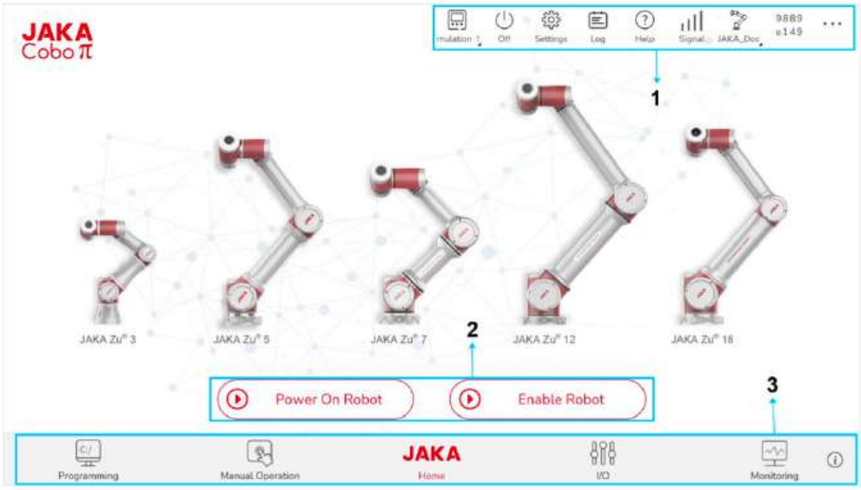


Figure 55 - JAKA APP home window.

Through the menu bar it is possible to switch between the real robot and the simulation, power down the control cabinet, set up the robot, connect/disconnect the robot network and show the log. Log records information, it can store historical data and diagnose problems. If an error alarm occurs during the operation, the user can look through the log information to find out the cause of the error and conduct inspection by clicking on the content of the message displayed in the log information interface. The messages could bring information about the state change of the robot, warnings when the robot is an abnormal state and errors that could cause program stopping or robot disabling. The switch bar is used to power on/off and enable/disable the robot, operations that can be manually executed also by using the control stick. When the ring-shaped light at the end of the robot changes into blue, the robot is powered on, while when the light changes to green means that the robot is enabled. The function bar can be used to create a program, move the robot by jogging, configure I/O functionalities and parameters and monitor the robot state. Some parameters as the temperature of the control cabinet, the current, the voltage, the torque, the speed limits for any of the six joints can be displayed by opening the monitoring window.

3.2.1 SETTINGS

The settings interface has 5 parts: system settings, operating settings, safety settings, program settings, and hardware & communication. During the MoMa prototype development, the focus has been centered on the operation dettings for a proper use of the system and on hardware & communication to define data exchange. When using the robot, there are various coordinate systems, including the world coordinate system, the end flange coordinate system, the tool coordinate system and the user coordinate system. The world and end flange coordinate systems are the default coordinate systems while the tool and user coordinate systems are defined if necessary. All of them use the right-hand rule as shown below:

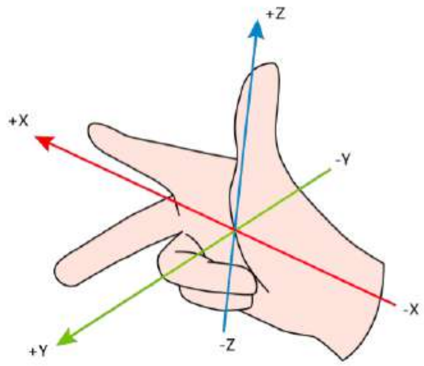
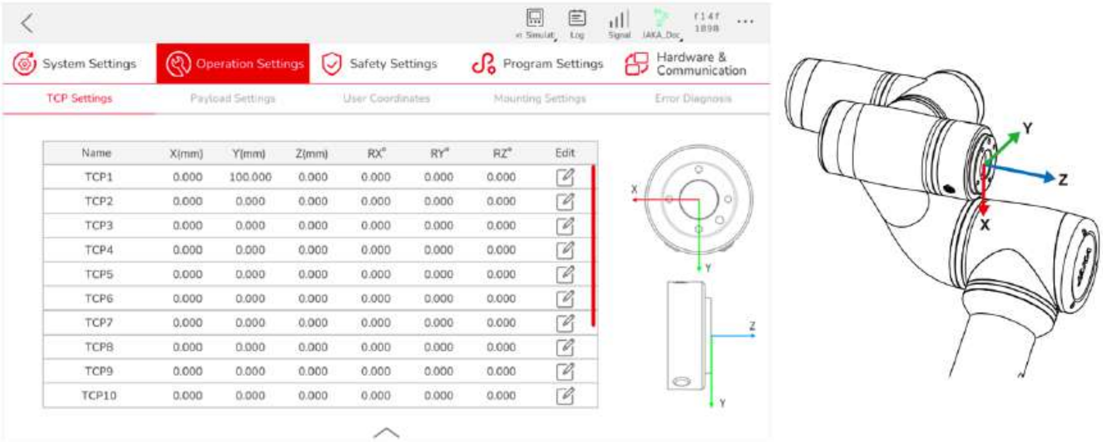


Figure 56 - Right-hand rule for coordinate system definition.

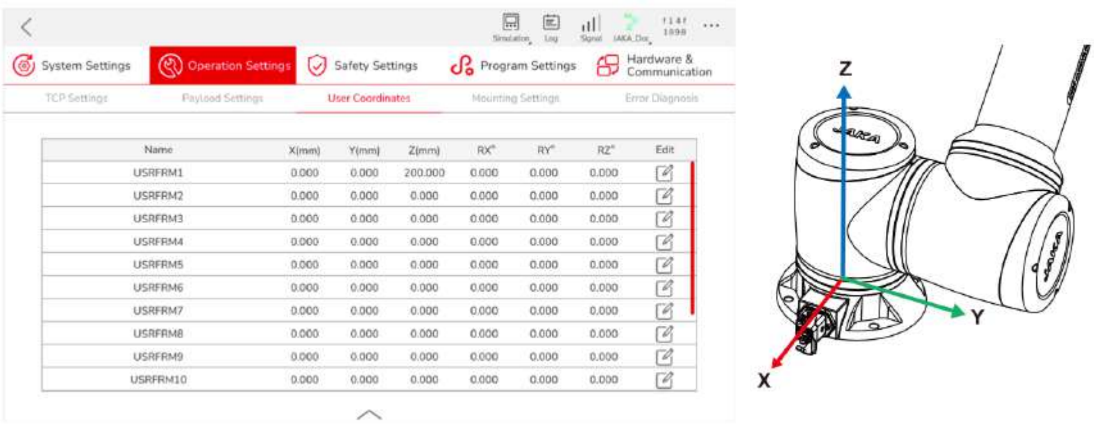
The TCP is the coordinate system established with the Tool Center Point (TCP) as the origin, which needs to be calibrated manually. It indicates the position of the robot tool. When the robot tool changes, it only needs to re-calibrate the tool coordinate system, while the positions within the robot program remain valid. The flange coordinate system is the default tool coordinate system. The origin of flange coordinate system is the center of the flange at the end of the robot, the flange facing outwards the positive direction of the Z-axis. In working conditions where high accuracy of robot movement is required, TCP is generally set at the end of the robot end-effector, such as in the center of the gripper or the suction cup. The JAKA APP provides 15 TCPs, the parameters of which are editable. The user can edit the TCP parameters by manually setting the input values of X, Y, Z, RX, RY, RZ or by using the four-point/six-point settings. The last two modalities are the ones used for tool calibration, starting from placing a fixed point in the space. Then, control the robot to reach this point with four/six different orientations. The end point of the TCP reaches that point and the desired pose transformation of the tool coordinate system relative to the flange coordinate system is automatically calculated. The desired TCP coordinate frame is calculated by the robot based on the results of the calibration.



Name	X(mm)	Y(mm)	Z(mm)	RX°	RY°	RZ°	Edit
TCP1	0.000	100.000	0.000	0.000	0.000	0.000	
TCP2	0.000	0.000	0.000	0.000	0.000	0.000	
TCP3	0.000	0.000	0.000	0.000	0.000	0.000	
TCP4	0.000	0.000	0.000	0.000	0.000	0.000	
TCP5	0.000	0.000	0.000	0.000	0.000	0.000	
TCP6	0.000	0.000	0.000	0.000	0.000	0.000	
TCP7	0.000	0.000	0.000	0.000	0.000	0.000	
TCP8	0.000	0.000	0.000	0.000	0.000	0.000	
TCP9	0.000	0.000	0.000	0.000	0.000	0.000	
TCP10	0.000	0.000	0.000	0.000	0.000	0.000	

Figure 57 - On the left, the JAKA APP interface for TCP creation. On the right, the end flange reference frame.

The user coordinate system is a coordinate system built on the workpiece and requires manual calibration. When the position of the workpiece changes, it is only necessary to re-calibrate the user coordinate system, while the positions within the robot program remain valid. The world coordinate system is the default coordinate system and its parameters cannot be modified. The world coordinate system is based on the center of the robot base as the origin. In the case of table mounting, the vertically pointing direction of the base towards the robot is the positive Z-axis direction. In addition, the JAKA APP provides 15 user coordinate systems with editable parameters if necessary. The user can edit the TCP parameters by manually setting the input values of X, Y, Z, RX, RY, RZ or by using the three-point settings. When using the last modality, the parameters in the X, Y and Z directions of the desired user coordinate system are calculated automatically by setting three points: the first point defines the origin, the second point identifies the positive direction of the X-axis while the third point determines the first quadrant of the XOY plane. The desired user coordinate frame is calculated by the robot based on the results of the calibration.



Name	X(mm)	Y(mm)	Z(mm)	RX°	RY°	RZ°	Edit
USRFRM1	0.000	0.000	200.000	0.000	0.000	0.000	
USRFRM2	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM3	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM4	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM5	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM6	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM7	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM8	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM9	0.000	0.000	0.000	0.000	0.000	0.000	
USRFRM10	0.000	0.000	0.000	0.000	0.000	0.000	

Figure 58 - On the left the JAKA APP interface for user coordinates creation. On the right, the the robot base reference

3.2.2 I/O CONFIGURATION

The MiniCab has 7 digital signals. When connecting the MiniCab, the I/O interface will display the actual signals, ensuring that a channel cannot be set as both DI and DO at the same time. The I/O interface can monitor the digital input/output state in the control cabinet. When the DO signal is triggered, the function will be activated. DI displays the state of its selected function in real time.

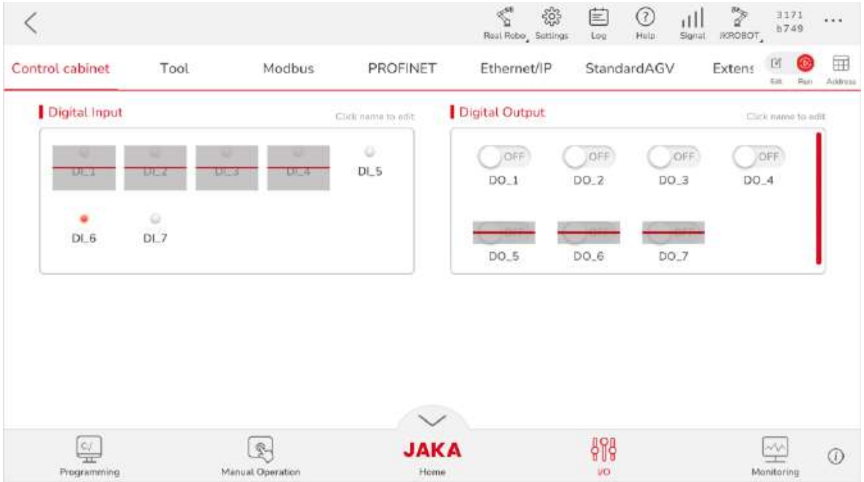


Figure 59 - JAKA APP window for control cabinet I/O assignment.

For the MoMa prototype, only 4 channels are addressed for digital signals, I/O assignment has been configured as reported in the picture. DO2 is used to control the opening/closing procedures of the bottom gripper. DO3 is used to manage the valve blower. Since positioning sensors are attached to the gripper, the cabinet receives DI5 and DI6 to capture the closed/open position of the gripper respectively. As can be seen from the image above, the gripper is normally open.

JAKA supports several communication protocols as Modbus, Profinet and Ethernet/IP. The most relevant for the MoMa prototype configuration is Modbus TCP/IP. The I/O signals in the "Modbus" interface are external I/O accessible by the robot. Instead of using the default window for Modbus I/O, two additional interfaces are created to manage the interaction with the I/O extension module and the AMR. The new Modbus connection must be configured by using the IP address and the access port. Both the configurations are reported below. For the extension module, 8 signals for digital input/output are defined respectively. For the AMR, 10 signals are defined, each one connected to a specific register address of the mobile robot.

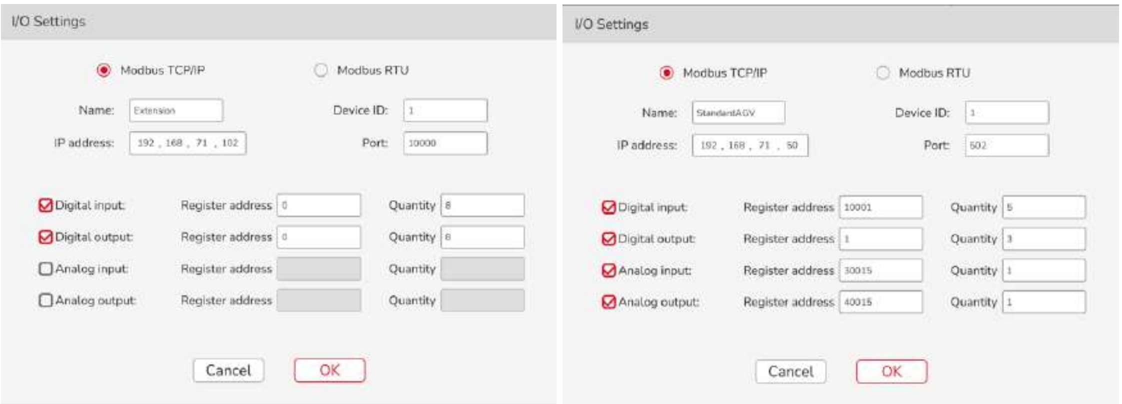


Figure 60 - On the left, the settings for Modbus I/O extension module connection. On the right, the settings for Modbus I/O AGV connection.

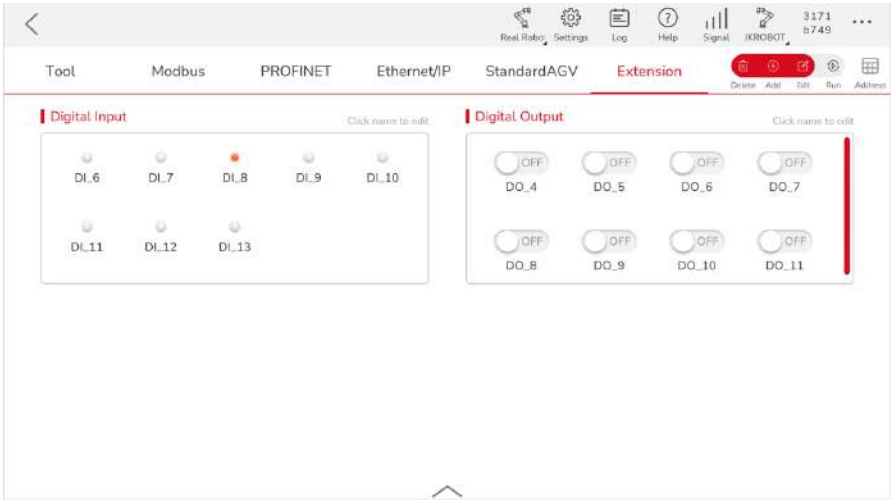


Figure 61 - JAKA APP window for Modbus extension module I/O assignment.

For the MoMa prototype, 10 channels are addressed for digital signals through the Modbus extension module, I/O assignment has been configured as reported in the picture. DO4 and DO5 are used to control the button with red and green backlight. DO6, DO7 and DO8 are used to control the color of a three-color led indicator, for red, yellow and green light signals. DO11 is used to control the opening/closing procedures of the side gripper. In this case the cabinet receives DI6 and DI7 to capture the pressure of the onboard buttons.

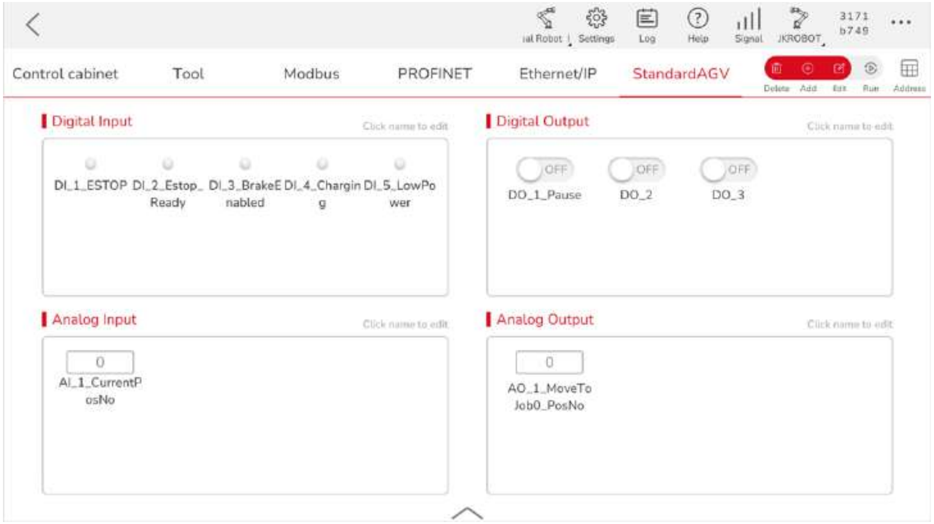


Figure 62 - JAKA APP window for AGV I/O assignment.

The communication between the cobot and the AGV is crucial for a good functioning of the system. Starting from looking at digital signals, 3 output are assigned starting from the register 00001, with the aim of pausing, continuing or stopping the movement. The 5 inputs are used to receive signals about the status of the mobile robot, as if the emergency stop is triggered or information about the charging of the battery. The analog signals are the only two signals used during the prototype development, since they are the one used to govern the navigation of the robot. The analog input associated to the register 30015 indicates the station in which the robot is located, reporting 0 if the current position is different from any station. The analog output associated to the register 40015, the same one used also for the test performed during the analysis of the mobile robot control, is the autonomous navigation signal. The overwriting of the station number in this variable will generate the motion of the MoMa toward the defined station.

3.2.3 MANUAL OPERATION AND PROGRAMMING

The JAKA APP supports manual operation interface to move the robot in real time. The movement can be executed with respect to two different coordinate systems: user coordinate or tool coordinate, both manually customized as explained above. From the top button "*Switch Coordinate System*" it is possible to shift between them, the red icon color shows the system in use. On both sides of the window, there is the option to control the execution of the movement, setting it as continuous mode or through step values.

The robot consists of six joints. Joint movement refers to the independent and manual control of each joint. Spatial movement, on the other hand, indicates the movement of the origin of the robot's tool coordinate system within Cartesian space.Users can choose whether to make movements in the user coordinate system or in the tool coordinate system. X, Y, and Z represent the translation positions of the robot flange center relative to the current user coordinate system. RX, RY, and RZ represent the rotational angles of the robot TCP relative to the current user coordinate system. The user must drag the slider on the left of the interface for spatial movement or the slider on the right of the interface for joint movement. The associated rotation of each joint is executed. Alternatively, manually set position and rotation values using the editor interface. At the bottom, a linear slider can be used to set the moving speed, expressed as percentage relative to the maximum value.

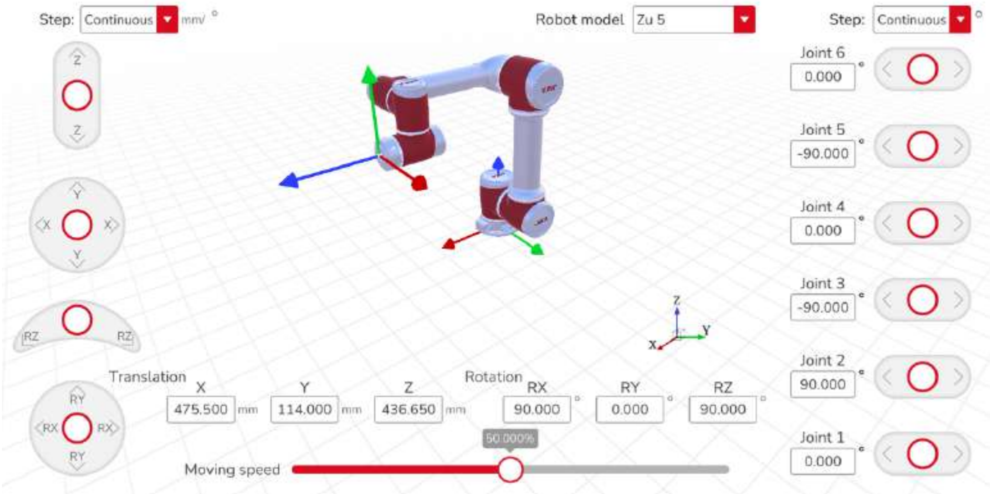


Figure 63 - JAKA APP manual control interface.

The JAKA APP provides an easy-to-use visual programming method, which greatly improves efficiency. Users can control the robot with a little programming knowledge. The interface is divided into three parts: A is programming commands, B is programming area, and C is program toolbar.

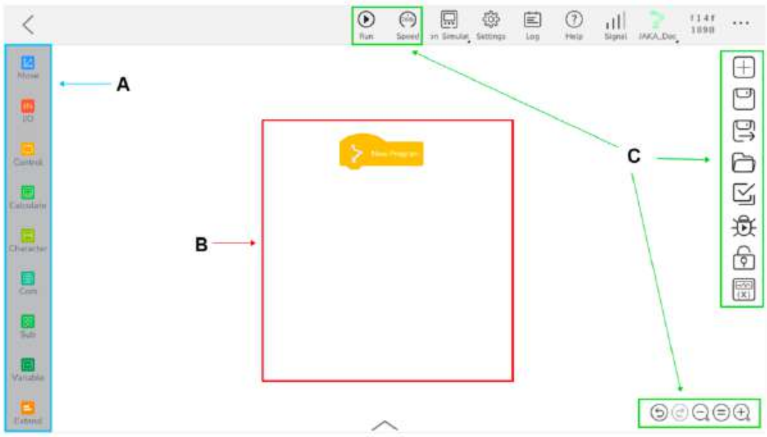


Figure 64 - JAKA APP programming interface.

The toolbar serves to monitor the execution of the program: run command, single-step debug, variable observation or speed controller, but also for other operations like save, import/export of the program. The programming commands are stored in the hidden menu on the left side, where the user can choose instructions to establish the actions that the robot must perform, to set input/output signals or to execute logic operations. The command blocks must be added to the programming area and placed according to the execution order.

3.3 JAKA LENS X

JAKA Lens X is a visual system using JAKA 2D camera, with the control system inserted in the robot control cabinet. The camera is equipped with JAKA self-developed visual operation software, the interface uses a web page to support cross-platform access. The new generation of machine vision software uses a fully graphical interface to introduce simplicity and powerful functions. Simplicity is given by the graphical, code-free interface, users do not need any professional programming skills, it is only required to configure parameters to easily complete the configuration of visual scenes. Powerful because the software can be used to generate several visual algorithm modules, which can be applied to any real scenario, improving operation efficiency. Users can complete vision applications including disordered grabbing, loading and unloading, palletizing, visual guidance positioning/assembly, defect detection, measurement and other advanced machines without writing code. The access to web interface for Lens X is provided by entering “control cabinet IP:1880”. The cross-platform access offers flexibility into the algorithm layer, any user can customize the functions to add, delete, or update items through the operation interface. Any user can also access the camera connected, change some of its parameters and customize the algorithm of the visual items and the parameters of each visual tool.

3.3.1 CAMERA CALIBRATION

As initial operation, before performing any test for the 2.5D alignment, it is necessary to execute the camera calibration, a one-time procedure required only when the camera installation changes.

The first step concerns the definition of the robot posture to ensure that the height and field of view meet the camera conditions. Once determined, the calibration board provided by JAKA must be placed in the camera view. The height distance recommended by the manufacturer for optimal accuracy is 200mm such that the chessboard occupies half of the camera view.

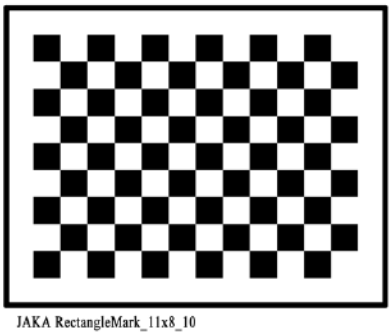


Figure 65 - JAKA calibration chessboard.

The Lens X calibration flow must be configured, starting from the definition of the port number for the calibration program socket (9999). In the camera management block, adjustments of the camera exposure and gain are required to obtain a focused image with appropriate brightness. In the calibration parameter block, the calibration name and type, the layout of the grid and the dimension must be set.

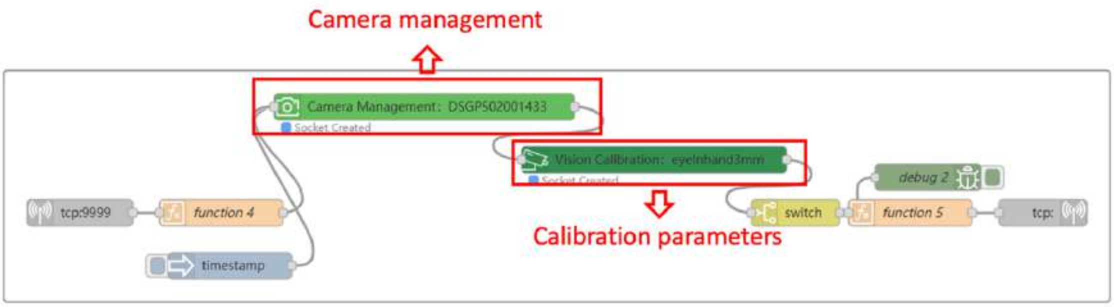


Figure 66 - JAKA Lens X calibration flow.

The following step involves the robot "*Calibration*" program. The user must save the capturing pose for the calibration (*calib_pos*) and specify the exact distance between the camera position and the calibration board (200mm). A socket must be established by using the IP address (192.168.71.26) and a port number (9999) to enable the communication between the camera and the MiniCab. Particular attention must be paid to the user and tool coordinates: the former defines the reference frame with respect to which the robot moves, the latter shall be adjusted throughout the program, choosing between the camera TCP and the null TCP. Since in this case TCP9 is not different from its default null definition, it has been selected for program purposes. After completing the setup, the "*Calibration*" robot program can be run and the eye-in-hand hand-eye calibration can be executed. This is the calibration method when camera is installed on the end of the robot and high precision is needed.

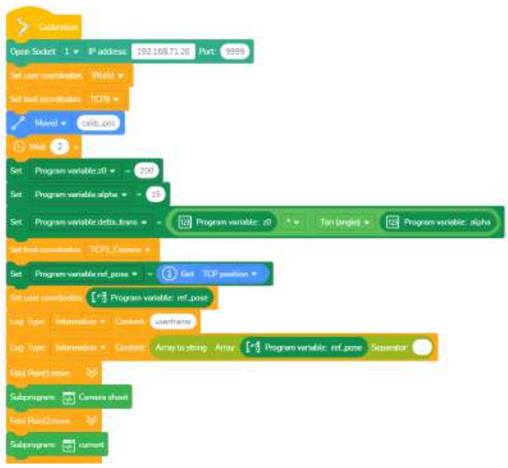


Figure 67 - Robot calibration program - initial settings.

Throughout the program, the robot moves in a sequence of 13 points around the initial pose. Each target point is defined with specific values of translations (*delta_trans*) and rotation angles (*delta*), used to set the coordinates values of the position array. Then, the robot moves toward the configured point.

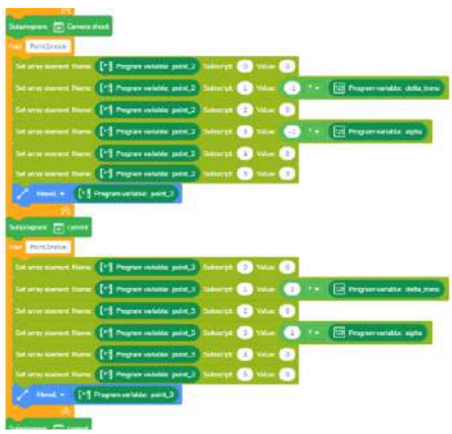


Figure 68 - Robot calibration program - details about coordinate configuration.

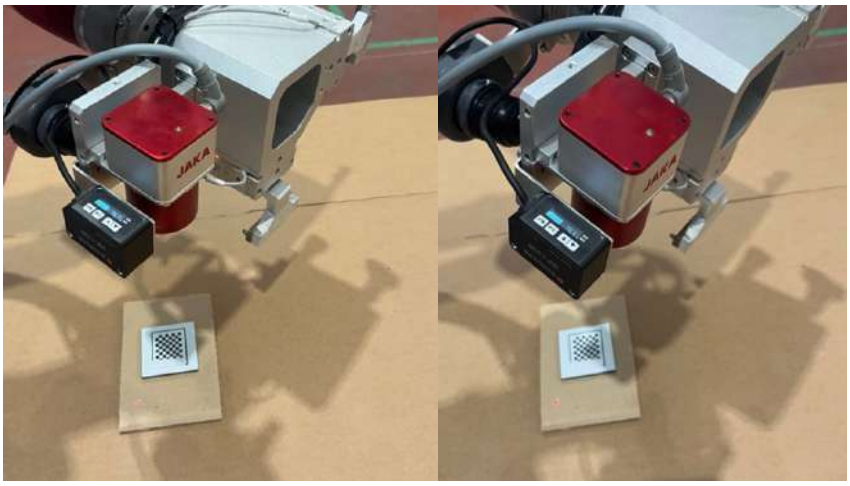


Figure 69 - Representation of two consecutive calibration position.

At each point, the robot briefly stops while "*Camera shoot*" subprogram is executed. The MiniCab sends a command to the camera via socket to acquire an image of the chessboard from that perspective. It is possible to monitor the progress during the execution through the Lens X interface, where pop-up messages appear to indicate whether the calibration has been correctly performed in the respective pose.

Figure 70 - CameraShoot robot program.

Once the robot finishes moving among the points, it returns to its initial pose. At this point, Lens X displays a final image for the calibration board, with results shown in the top-left corner. The Camera Matrix RMS measures the accuracy of the camera intrinsic parameters (focal length, principal point), with lower values indicating better calibration. The optimal value is typically below 1 pixel. The Extrinsic RMS measures the accuracy of the camera position and orientation in space relative to the world, with lower values indicating better alignment. The optimal value is also as low as possible for precise calibration. If the calibration is completed successfully, the calibration file will be saved and used for the following 2.5D localization setup.

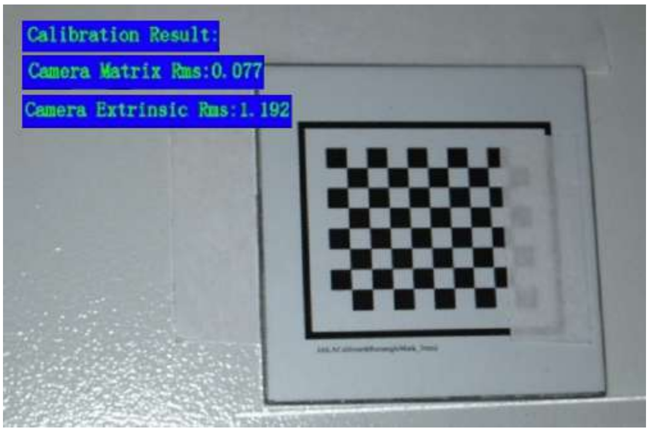


Figure 71 - Calibration result.

CHAPTER 4 – ACTIVITIES PERFORMED AND PROTOTYPE VALIDATION

The introduction of Mobile Manipulators has increased over the years in industrial environments for various scenarios: CNC machine loading/unloading, semiconductor wafer cassette transfer, automatic wafer cassette loading/unloading, inter-process FOUP transfer, auxiliary material delivery, drug transfer between multiple stations and electrical cabinet switch inspection.

Industrial robots are programmable machines capable of executing repetitive operations. They are usually installed in a fixed position, operating in a known environment that enables precise operations. When a manipulator is mounted on a mobile robot, the accuracy of the manipulation strictly depends on the positioning of the AMR at the working station. Many factors must be considered during the navigation, such as sensor uncertainties and wheel slippage. The precise location is only partially known, as localization techniques are accurate within a margin of a few centimeters. Therefore, the cobot may be placed differently with respect to the working area. The introduction of a 2D vision system may increase the overall efficiency.

The 2.5D alignment presented in this work is a technique developed by JAKA to compensate for mobile robot positioning by introducing a marker-based visual servoing. The task of visual positioning is a vision-based feedback system to control the robot pose by observing the marker. This strategy allows the robot always to reach the same position, mapping the working area as defined with respect to the marker. Once the mobile robot approaches the station, the cobot aligns itself with the marker attached at a predefined location and, after the alignment, executes its movements. The position of the working points relative to the robot are determined by tracking the marker location.

The most technical advantage of this strategy is the lower cost compared with 3D camera. According to the manufacturer value proposition, the 2.5D alignment technology offers positioning accuracy up to ± 0.5mm in spatial pose and faster takt time compared to similar competitors within 2s, even if the captured image is blur.

The final goal of this work is to validate the overall functionality of the system. For this purpose, a series of tests have been conducted. This section begins with an explanation of the 2.5D alignment method. Next, static additional tests are presented to evaluate the overall system accuracy. Finally, a simulation of a practical real-world industrial application has been developed, including the motion of the mobile robot.

4.1 2.5D ALIGNMENT PROCEDURE

This phase consists of two main processes, which are called "*Debug mode*" and "*Online mode*". All the activities are performed supposing that the calibration has been already executed, as reported at the end of the previous chapter. Even the 2.5D localization procedure is a one-time procedure required only when the camera installation changes. The main scope of this technique involves the definition of a reference position to which the cobot always aims to return when the MoMa approaches the workstation: this position must be generated by marker alignment.

Starting from defining the robot pose, ensuring that the camera field of view is centered on the *AlignMark* provided from JAKA. The height distance recommended by the manufacturer for optimal accuracy is 200mm such that the marker occupies half of the camera view. This capture position depends on the application, can be different from the calibration position.

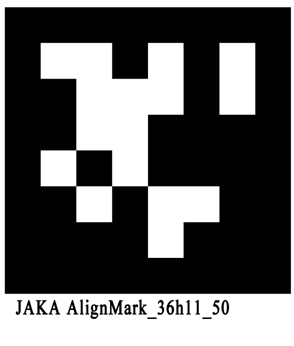


Figure 72 - JAKA AlignMark.

Also in this case, settings regarding the exposure and gain in the camera management block must be adjusted.

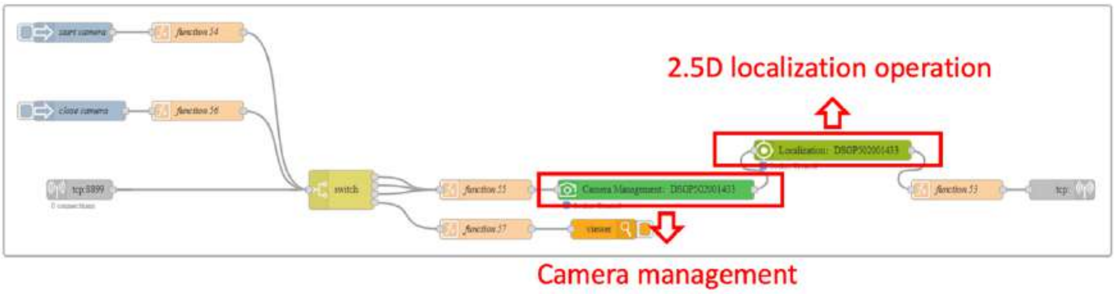


Figure 73 - JAKA Lens X 2.5D calibration flow.

Before moving forward with the testing procedures, proper tuning of these parameters needs to be performed for the environmental light conditions, to ensure a focused image. In the following pictures, it can be observed that exposure value of 70000 and a gain of 3 were the optimal settings to achieve a clear view of the marker during the testing phase.

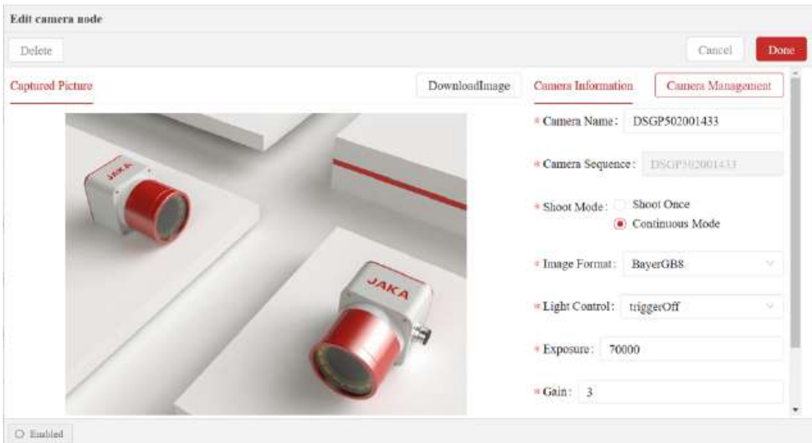


Figure 74 - JAKA Lens X camera management for parameters setting.

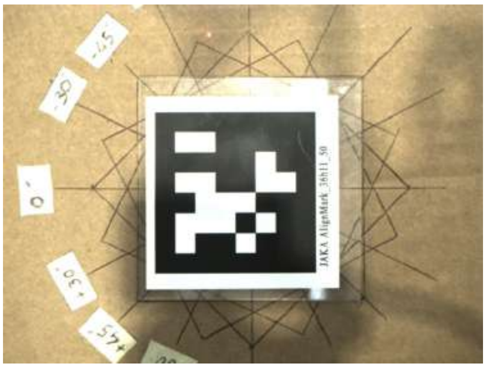


Figure 75 - Camera view used for alignment procedures.

The following configuration involves the robot "*MarkerLocalization*" program. The user must save the debug pose for the marker localization (*DebugPosition*) and the user and tool coordinates, leaving *World* and *TCP9* as default settings. The "*Debug*" folder contains code developed by the manufacturer, which has been provided with proper explanations of its purpose and usage. The internal content of this folder is not analyzed, as it is not relevant for the objectives of this work. Even the definition of the parameters "*corner_thresh*" and "*loc_num_thresh*" has not been modified and remains as provided by the manufacturer. These values are configured for set up an optimal threshold for the alignment process, smaller values may not allow the robot to achieve the adjustments.

Figure 76 - Marker Localization robot program.

Since the coordinates of the "*DebugPosition*" with respect to the robot base are known, it is useful to setup a new userframe, called "*USRFRM2_NullTCPDown*" with these coordinates. In this way, this position can be considered as the origin of the map of the working area where the cobot will operate. The coordinates of the flange expressed with respect to this reference frame are null.

Figure 77 - Representations of the "DebugPosition": on the left, with coordinates of the TCP expressed with respect to the robot base, on the right the coordinates of the TCP expressed with respect to the new userframe.

USRFRM2_NullTCPDown	453.000	117.000	317.000	90.000	0.000	90.000

Figure 78 - New coordinate frame configured with the same coordinates as the "DebugPosition".

This position will serve as the reference pose that the robot, guided by the camera, aims to reach after each alignment with the marker. The *AlignMark* in Lens X 2.5D flow must be captured, specifying its parameters, the TCP ID and referring to the calibration file generated as reported at the end of the previous chapter. Once the pose is saved, the Lens X interface will display the grayscale version of the marker view, the image that the camera will aim to match by moving the robot to the same pose relative to the marker.

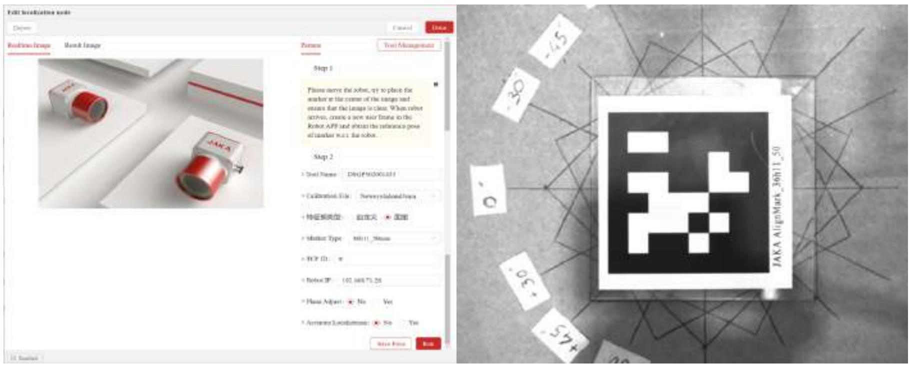


Figure 79 – On the left, JAKA Lens X camera management for 2.5D localization settings. On the right, the image captured by the camera.

After completing the setup, run the robot program and the camera localization flow to automatically execute the marker localization. Therefore, if the localization is successful, the robot program can be stopped. Once the procedure is completed, in the same window the Lens X interface will show the result of the localization.

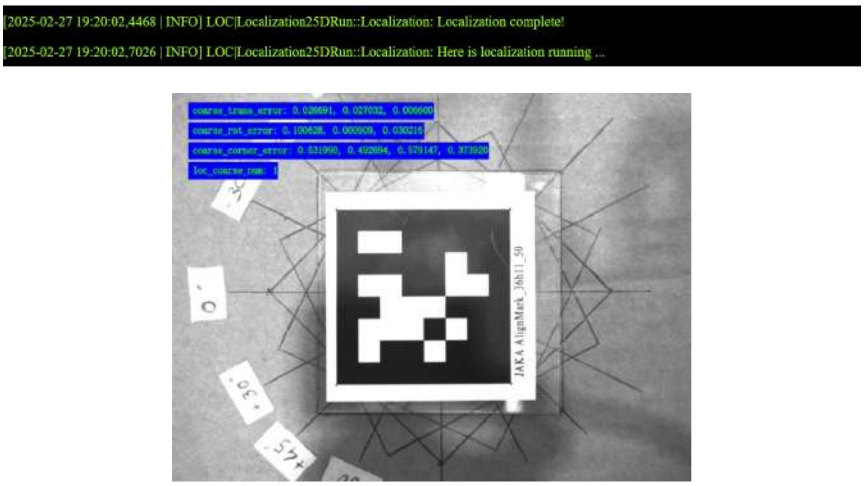


Figure 80 - On the top, the notificationof completed localization displayed in the Lens X log block. On the bottom, the results of the localization.

The upper left corner returns the theoretical positioning error: "*coarse_trans_error*" is the translation deviation of the feature board in the camera coordinate system, "*coarse_rot_error*" is the rotation deviation of the feature board in the camera coordinate system, "*coarse_corner_error*" is the pixel deviation of the feature board in the camera coordinate system. The last information "*loc_coarse_num*" is the number of robot adjustments in "*Debug mode*". Finally, click on "*Done*".
At this point, it is possible to teach the movement to the target, setting as reference frame the userframe previously configured and TCP9. As the initial setup for the tests performed, the motion taught to the robot is such that it aligns the laser, mounted alongside the camera, to a corner of the *AlignMark*.

The "*Online mode*" is the operation that executes the alignment process. In this phase, a specific robot position, usually referred in the following tests as "*ShootPosition*", can be defined. This is the pose from which the camera starts searching for the marker, ensuring that the marker is within the camera field of view before initiating the alignment corrections. The "*Online*" folder contains code developed by the manufacturer, accompanied by explanations regarding its purpose and usage. The internal content of this folder is not analyzed, as it is not relevant to the objectives of this work. Within this folder, a connection to the port number for the 2.5D localization program socket (8899) is defined. This is the section where the robot makes slight adjustments upon receiving position feedback commands from the camera, aligning the marker view with the reference scene defined earlier. The localization results are reported in the Lens X interface.

Once the alignment procedure is concluded, the cobot is in a known position with respect to the working area, with some errors due to the marker localization. If the mobile robot stops with a certain inaccuracy with respect to the working station, the manipulator will be in a different relative position to the working space. Performing the movement recorded before, the instruction will not bring the robot to the target. The JAKA APP programming interface allows to set as user coordinate the actual tool position and perform a relative movement based on it. At this point, performing the movement, the cobot will reach the target.

Figure 81 - Programming instruction that stores the actual position of the TCP as user coordinate frame.

This command is crucial to store the marker aligned pose and move relatively to it. Once the program is launched, this userframe has null coordinates that after the "Online mode" will be overwritten by the values of the actual position of the TCP. From the manufacturer instructions, this command only influences the *MoveL* and has no effect on the *MoveJ*. This completes the visual servoing procedure: starting from a 2D calibration board, it sets a 3D reference frame at the flange of the cobot when it is aligned within the working space and move relatively to it.

4.1.1 ORIENTATION TEST

This main purpose of this test is to verify the alignment of the manipulator according to different orientation of the marker, considering rotation about all the axes. This test is carried out statically: the mobile robot is fixed in a position, the robotic arm starts from a "*Home*" position, then it moves to the "*ShootPosition*" to execute the alignment. The *AlignMark* is attached on specific supports set at various angles to test the alignment about RX, RY and RZ rotations. Starting from considering RY, which corresponds to the vertical axis of the camera for the MoMa layout. The base shows various rotation angles around this axis, as illustrated in the figure, on which the marker will be attached.

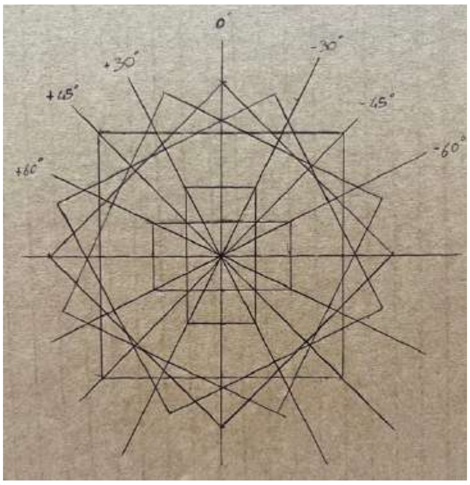


Figure 82 - Base configuration for RY Orientation test.

Due to space constraints related to the working station setup, the test has been performed for rotations from -60° to 60° degrees with defined intervals. The success of the alignment can be verified by observing the robot position, which will show rotations around RY corresponding to the board orientations. It is possible to evaluate the robot rotations with respect to the userframe defined in the "*Debug mode*". Additionally, some translations of the robot are required to align the camera with the marker. Due to the camera mounting position, the robot flange effectively perform the alignment rotating about the camera rather than about its own axis.

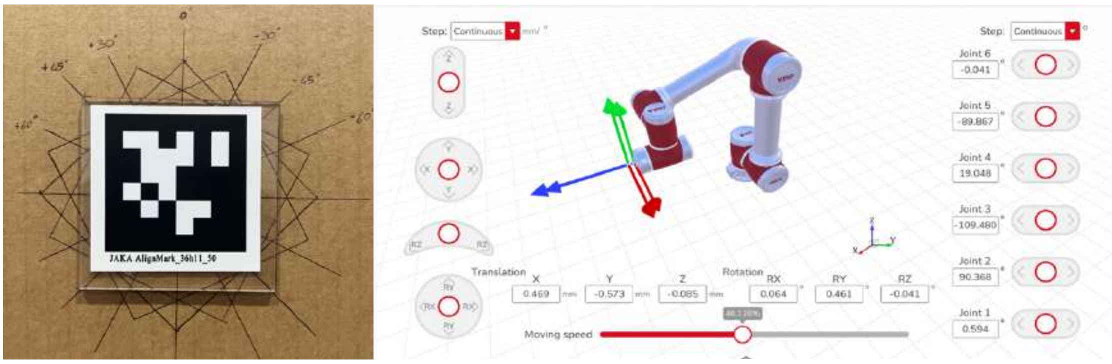


Figure 83 - CASE 0°. On the left, the marker orientation. On the right, the relative robot alignment.

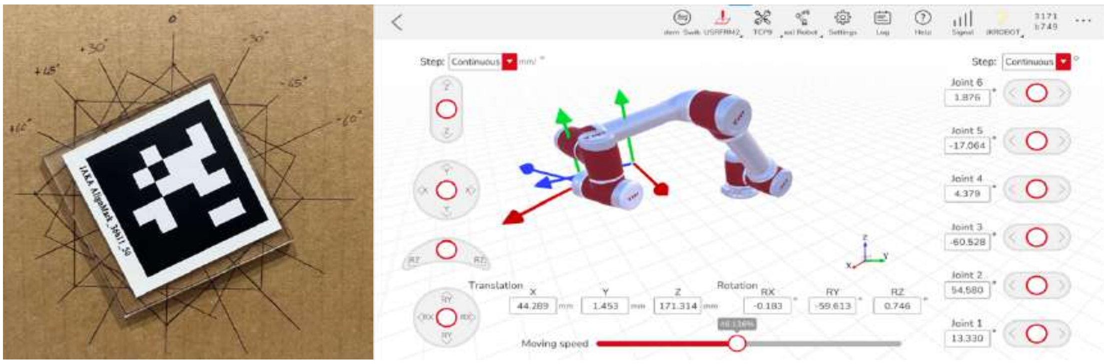


Figure 84 - CASE -60°. On the left, the marker orientation. On the right, the relative robot alignment.

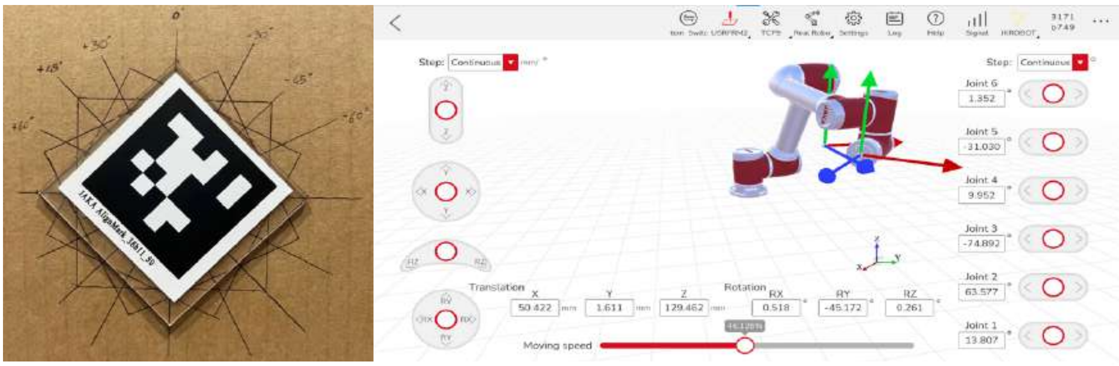


Figure 85 - CASE -45°. On the left, the marker orientation. On the right, the relative robot alignment.

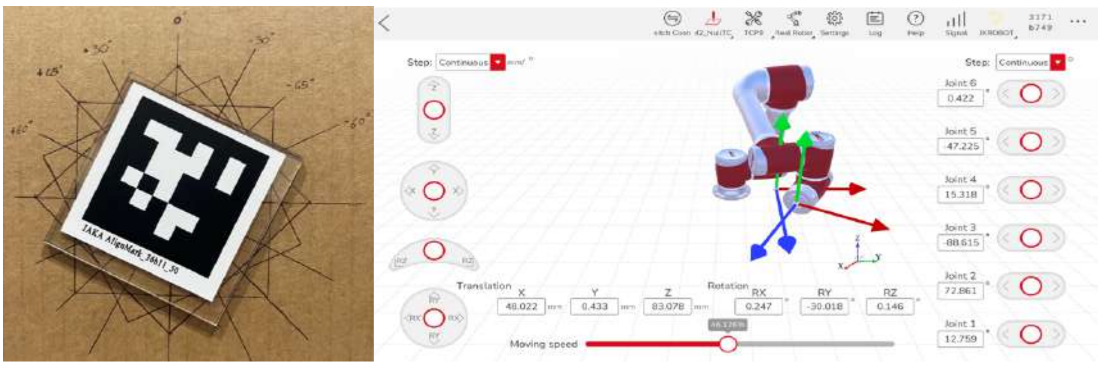


Figure 86 - CASE -30°. On the left, the marker orientation. On the right, the relative robot alignment.

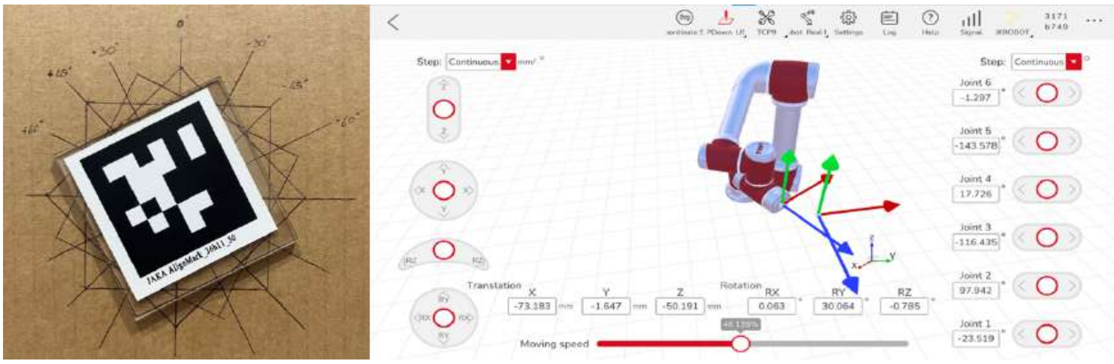


Figure 87 - CASE 30°. On the left, the marker orientation. On the right, the relative robot alignment.

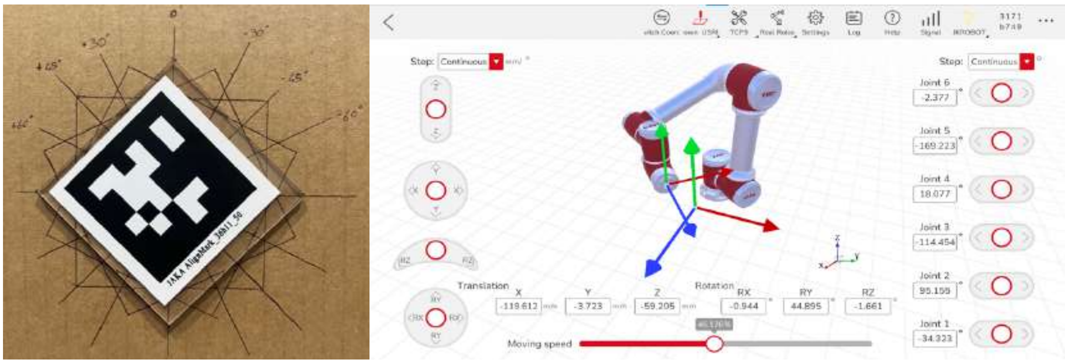


Figure 88 - CASE 45°. On the left, the marker orientation. On the right, the relative robot alignment.

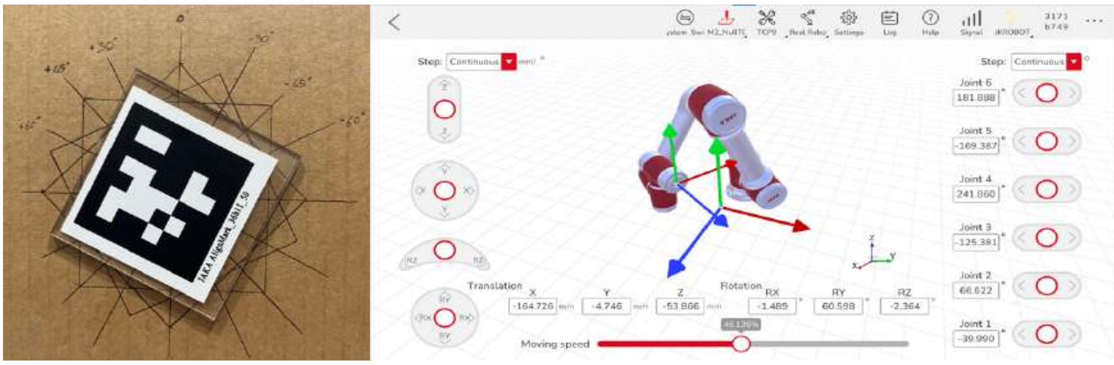


Figure 89 - CASE 60°. On the left, the marker orientation. On the right, the relative robot alignment.

The same concept is also developed for rotations about RX, RZ. The marker is attached to a specific support with a 30° of inclination. In this case, instead of showing the detailed position of the robot, some examples of real alignments are reported.

Figure 90 - CASE RX 30°. On the left, the robot in ShootPosition. On the right, the aligned pose of the robot.

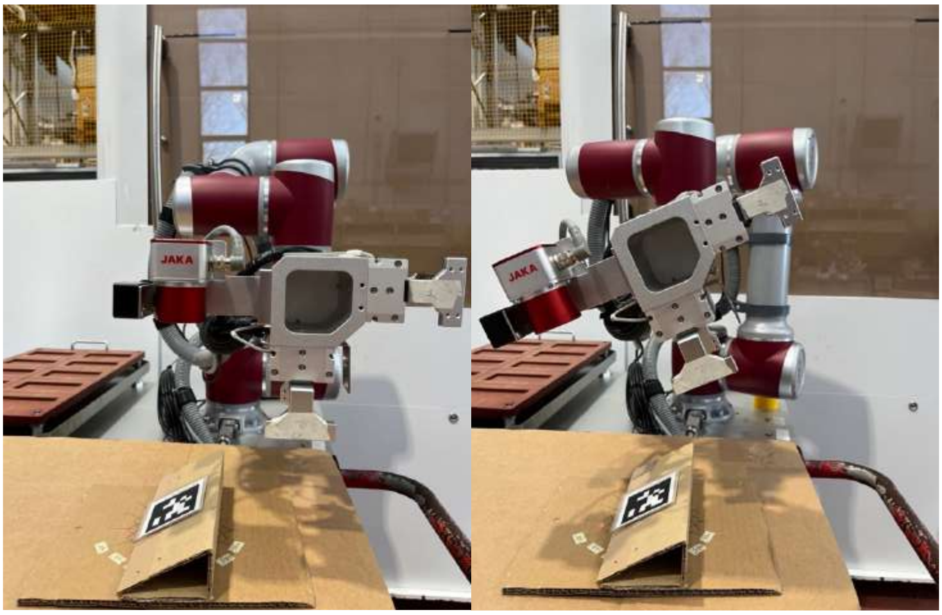


Figure 91 - CASE RZ 30°. On the left, the robot in ShootPosition. On the right, the aligned pose of the robot.

Looking at the results, it can be observed that the camera successfully aligns the position of the flange in a parallel plane to the one where the marker is placed, maintaining the constant distance as configured. To test also the relative positioning based on the alignment, the commands to execute the motion saved during the "*Debug mode*" are introduced in the program. However, these tests are not sufficient to fully validate the accuracy during the alignment procedure, since the marker is manually attached to the base. This operation, combined with uncertainties in the construction of the reference direction of the bases, may impact the alignment accuracy.

Figure 92 - Orientation test robot program.

4.1.2 RECOGNITION TEST

This test has been conducted to verify the distances at which the camera can detect the marker. Starting from the *ShootPosition* placed at 200mm from the marker, a simple program with a loop cycle has been implemented. For each execution, a counter increases the vertical distance from the marker of 100mm, until the maximum distance limit imposed by the robot construction is reached. In the final cycle, instead, the robot is moved closer to the marker to verify whether the camera can still recognize it. As in the previous test, at the end of each cycle, the robot performs the predefined movement, pointing the laser toward one of the marker's corners.

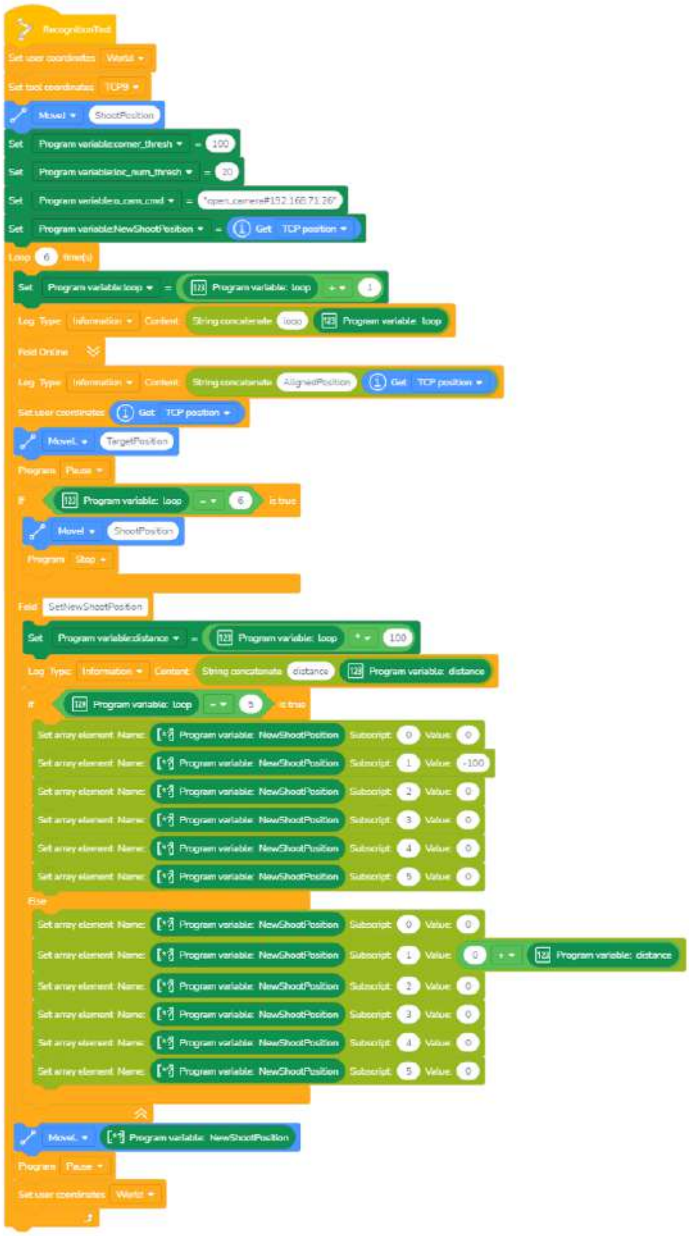


Figure 93 - Recognition test robot program.

This test has been executed at distances between 100mm and 600mm from the marker, in 100mm intervals. The camera consistently localizes the marker even when it is blur, making the necessary positioning adjustments to align the robot with the marker position. As one might expect, the variation in distance impact the number of adjustments the robot needs to make to reach the desired pose: shorter distances require less adjustments than longer distances. All the tests have been performed consequently under the same light conditions.

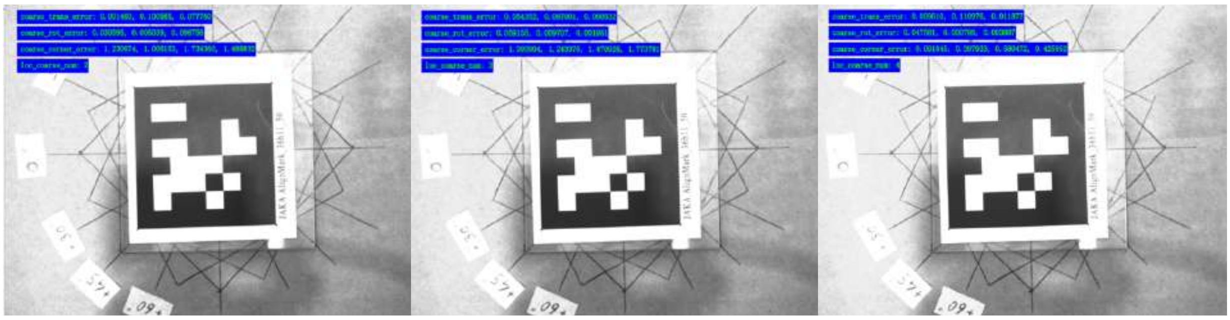


Figure 94 - The results of the alignments in the Recognition test. From the left, 100mm, 200mm, 300mm.

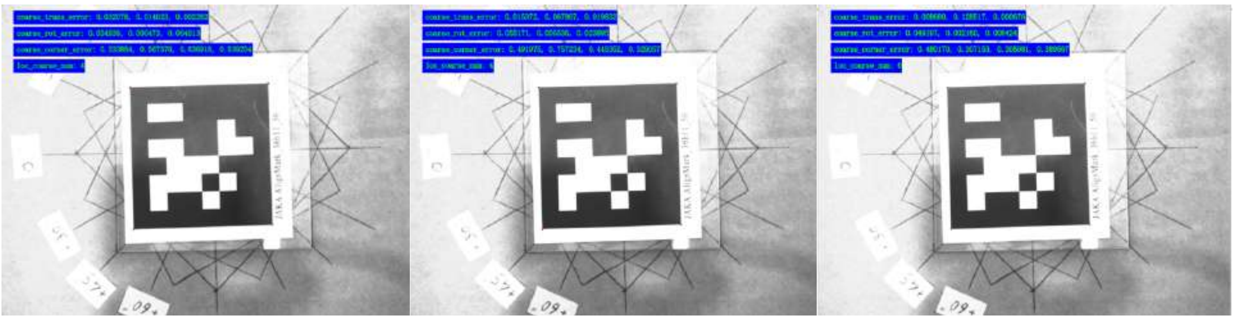


Figure 95 - The results of the alignments in the Recognition test. From the left, 400mm, 500mm, 600mm.

4.1.3 RANDOM TEST

The idea behind this test is a combination of the previous two. Until this point, the "*ShootPosition*" defined in the tests was the same as the "*DebugPosition*" used for the alignment. In this test, however, the position is assigned randomly at any distance from the surface where the marker is placed. The robot has been manually moved in free-drive mode, with the only requirement being that the marker must be within the camera field of view. Additionally, the *Orientation Test* previously presented only verifies the alignment for a specific rotation of the marker, whereas in this case, the marker is placed randomly. Some alignment tests conducted under these conditions, with different starting positions, are shown in *Figure 97*.

Figure 96 - Recognition test robot program.

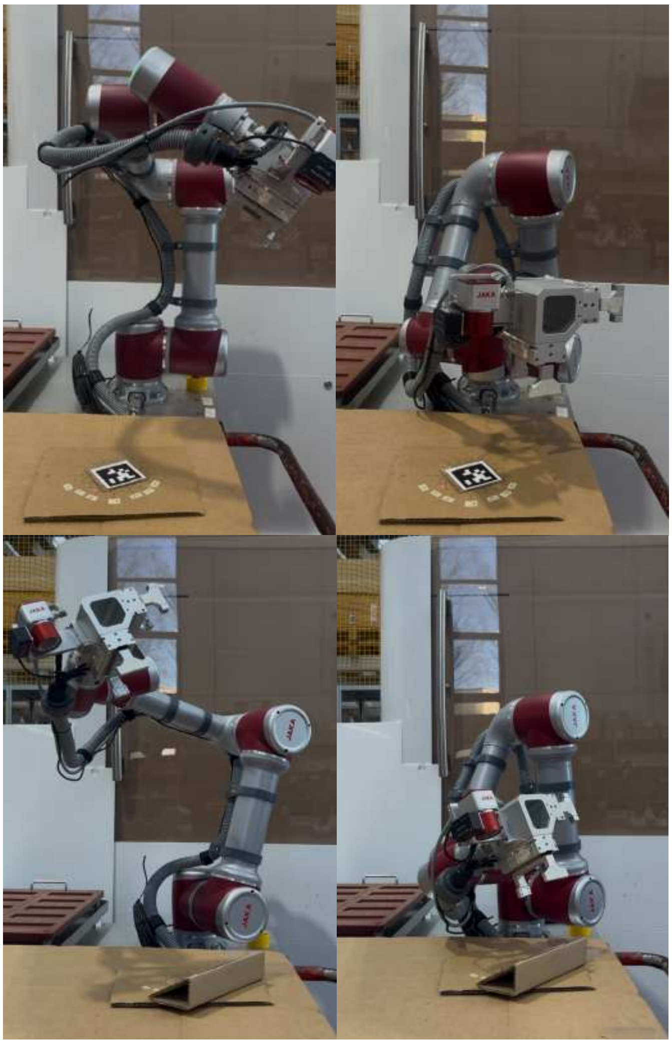


Figure 97 - Two cases of Random test. On the left, the initial Shoot Position of the robot, on the right the robot pose after the alignment.

4.1.4 X-Z ERROR TEST

While in the previous *Recognition Test* the robot has been only moved away from the marker along the vertical direction, in these tests the translations along the X and Z axes have also been performed. This approach allows verifying the alignment under all possible positioning uncertainties introduced by the mobile robot, along any axis.

In the case in which the marker is placed on a horizontal plane, the AMR navigation inaccuracies directly impact on the X-Z plane. If the marker is attached on a vertical plane, the distance from the marker depends on the AMR positioning, which will impact on the X-Y plane. These tests are designed in parallel, with only minor differences between them.

Suppose to mention only the "*PositioningXErrorTest*" which focuses on the X-axis. The program consists of a loop that repeatedly translates the "*ShootPosition*" by 10 mm in both the directions along the X-axis. Once the maximum possible translation is reached, beyond which the camera is no longer able to detect the marker, the program increases the height to start a new set of translations. The loop consists of 10 cycles, meaning that a possible positioning error of 100 mm along the X-axis is analyzed. This uncertainty is larger than the typical errors generated by the mobile robot navigation. However, even assuming such a large error, placing the robot at 500 mm away from the marker allows the system to successfully perform the alignment, since the camera is still able to detect the marker at this distance. The same reasoning is at the basis of the dual program for the Z-axis translation.

In the end, what these tests are spatially reproducing is a 3D volume where the marker occupies one face and other faces are determined according to the translation and heigh adjustments performed in each test. This volume acts as detecting zone, ensuring that the marker remains within the camera view for successful detection and alignment.

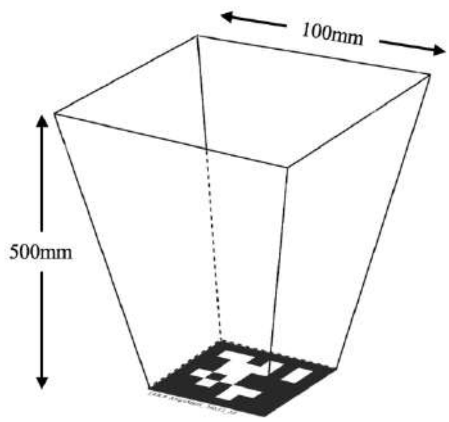


Figure 98 - Volume in which the marker is succesfully detected by the camera.

4.2 PERFORMANCES EVALUATION

Nowadays, two measures are commonly used to describe the positioning performances of industrial robots: repeatability and accuracy. Each of these quantities depends on the various components used in constructing the robot (motors, encoders, links) [30].

Repeatability is the ability of the robot to return repeatedly to the same pose. This parameter can be affected by lost motion (hysteresis, backlash, and torsional elasticity) and friction in the gear trains, but also thermal expansions must be considered when the system works for long periods of time.

Accuracy is the ability of the robot to precisely move to a desired position in 3D space, mostly influenced by geometric inaccuracies and elasticity, present in both the links and the transmissions [31].
Those parameters can also be considered to evaluate the MoMa alignment performances. Two new tests have been introduced in the following sections to verify if the accuracy of the 2.D localization technique correspond to the manufacturer proposition.

4.2.1 ALIGNMENT PRECISION TEST

For all the previous tests, the focus was centered on the correctness of the alignment operation, without considering any accuracy in the obtained results. In this case, the attention has been devoted to the movement of the cobot in the space.

The idea behind this test is to save the coordinate values of the "*ShootPosition*" and then execute the alignment 20 times. After each operation, the aligned position is captured and showed in the log interface as an information. Then, as usual, the robot moves toward the target through the movement previously taught. This test has been executed at 200mm and 400mm of distance from the marker, to compare the values and highlight if the distance from the marker may have impact on the accuracy.

Figure 99 - Alignment Precision test robot program.

All the data, with the reference position and the 20 aligned position computed at different heigh are reported in *Table 7* shown below. At the end, three rows display the maximum error, the minimum error and the average error obtained from alignments with respect to the reference pose.

	X	Y	Z	RX	RY	RZ
Shoot Position H200	453,000239	117	317,000305	89,99996	0	89,99999
Aligned Position 1	454,408814	118,524206	317,648328	90,105557	-0,113515	89,31429
Aligned Position 2	453,795742	118,253679	317,421709	89,96147	-0,164832	89,319227
Aligned Position 3	454,489408	118,079688	317,883782	90,129598	-0,232526	89,321632
Aligned Position 4	454,294332	118,122953	317,750536	90,087092	-0,197131	89,315352
Aligned Position 5	454,536127	118,429565	317,731619	90,132489	-0,139906	89,31319
Aligned Position 6	454,084212	118,02927	317,673443	90,042184	-0,218646	89,322685
Aligned Position 7	454,139455	118,178277	317,623276	90,04883	-0,184015	89,316441
Aligned Position 8	454,164792	118,183302	317,64844	90,060543	-0,184978	89,321434
Aligned Position 9	453,887491	118,306637	317,418032	89,982743	-0,154993	89,314236
Aligned Position 10	454,396619	118,215846	317,74446	90,109807	-0,172699	89,314295
Aligned Position 11	454,122008	118,214897	317,594551	90,042115	-0,174377	89,317933
Aligned Position 12	453,938401	117,993982	317,569885	90,001748	-0,247897	89,327412
Aligned Position 13	453,914553	118,041539	317,524539	89,991422	-0,239915	89,325162
Aligned Position 14	454,339307	118,396682	317,856763	90,181128	-0,151129	89,30943
Aligned Position 15	454,348951	118,164978	317,728593	90,097683	-0,188055	89,31134
Aligned Position 16	454,164398	118,080108	317,678383	90,056943	-0,207932	89,315631
Aligned Position 17	454,229671	118,012242	317,740396	90,077338	-0,227998	89,318622
Aligned Position 18	454,283331	118,307977	317,604125	90,079127	-0,17065	89,315673
Aligned Position 19	454,62531	118,03014	317,979634	90,175791	-0,227189	89,319224
Aligned Position 20	453,977745	118,127085	317,551686	90,004247	-0,197473	89,316473
MaxError	1,625071	1,524206	0,979329	0,181168	0,113515	0,672578
MinError	0,795503	0,993982	0,417727	0,03849	0,247897	0,69056
AverageError	1,210287	1,259094	0,698528	0,109829	0,180706	0,681569

	X	Y	Z	RX	RY	RZ
Shoot Position H400	452,999947	117	517,000002	90,00005	0	90,000001
Aligned Position 1	453,943885	118,453865	517,319444	89,990467	-0,117287	89,310235
Aligned Position 2	454,684103	118,324825	517,825957	90,190852	-0,153693	89,313057
Aligned Position 3	453,87143	118,473764	517,259594	89,972186	-0,112847	89,305876
Aligned Position 4	453,983068	118,363591	517,329202	90,004584	-0,137043	89,308818
Aligned Position 5	453,692913	118,20784	517,299038	89,933839	-0,175457	89,31243
Aligned Position 6	453,772335	118,15098	517,391613	89,954644	-0,188042	89,317149
Aligned Position 7	453,876239	118,229424	517,36855	89,971125	-0,174959	89,303823
Aligned Position 8	453,950867	118,078528	517,521084	90,002559	-0,206105	89,315473
Aligned Position 9	454,295498	118,35965	517,540924	90,073206	-0,138736	89,2999938
Aligned Position 10	454,340263	119,040804	517,385152	90,110822	0,019894	89,317605
Aligned Position 11	453,888933	118,115603	517,422856	89,982394	-0,190102	89,313858
Aligned Position 12	454,056728	118,175288	517,53134	90,029237	-0,186091	89,315939
Aligned Position 13	454,463748	118,134296	517,766916	90,139975	-0,198739	89,319033
Aligned Position 14	454,045979	118,440389	517,34752	90,012102	-0,118116	89,304257
Aligned Position 15	454,631799	117,977988	517,963481	90,173764	-0,236255	89,314295
Aligned Position 16	453,669806	118,468128	517,093765	89,925984	-0,111198	89,307073
Aligned Position 17	453,930411	118,075199	517,517094	89,996033	-0,208533	89,313656
Aligned Position 18	453,530398	118,183797	517,207823	89,894813	-0,182401	89,317146
Aligned Position 19	454,101739	118,373714	517,41656	90,02994	-0,136965	89,30541
Aligned Position 20	454,549464	118,149274	517,800406	90,149254	-0,19272	89,306156
MaxError	1,684156	2,040804	0,963479	0,190802	0,019894	0,680968
MinError	0,530451	0,977988	0,093763	0,105237	0,236255	0,7000072
AverageError	1,1073035	1,509396	0,528621	0,1480195	0,1280745	0,6904876

Table 7 - Results of Precision alignment tests. On the top at H200 mm, at the bottom at H400mm.

Looking at the results, it is possible to reflect on the accuracy of the alignment system. As can be noticed from the alignment errors, the distance from the marker does not significantly affect accuracy, since the uncertainty remains almost constant.

Analyzing the obtained results, it becomes evident that the adopted configuration did not achieve the precision specified by the manufacturer. However, this level of accuracy is still acceptable for some industrial applications such as logistics, palletizing and packaging. For more precise task, including automotive precision assembly, CNC positioning, handling of electronic parts and mechanical operations on objects, tolerances are stricter. A new calibration process and 2.5D localization, with optimized settings for camera parameters, may improve those results. Nevertheless, additional devices or systems could be introduced to enhance precision, such as mechanical centering systems, guiding fixtures or alignment aids which can be directly installed in the working space.

4.2.2 DISPLACEMENT PRECISION TEST

The second test aimed at evaluating performances focuses the motion. It may be interesting to observe how the alignment error influences subsequent movements. In this case, instead of considering the movement instruction saved in the debug mode, some new procedures have been defined. An alternative method had to be found to estimate the position accuracy, since the laser pointer beam is often wide and its size varies depending on the height from which it is projected. Specifically, the idea has been to perform a “drawing task” in which the robot first aligns with the marker, then moves into a defined position and slowly performs a linear vertical movement to gently touch the target.

Figure 100 - Displacement precision test robot program.

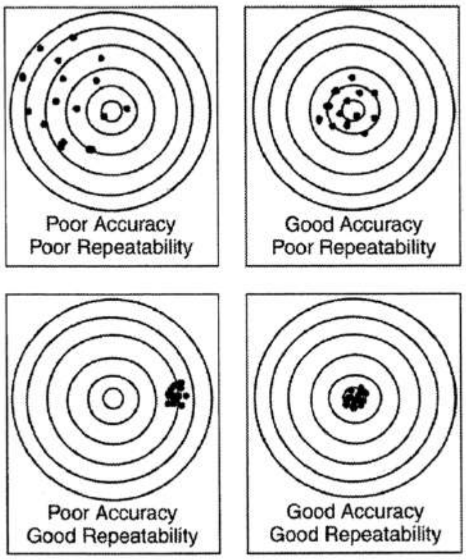
For this purpose, it has been required to attach a marker at the gripper: a fine-point marker have been chosen, to ensure the maximum accuracy during the drawing task. The key factor to consider for this test is the stability of the marker. If the marker shift position during the movement of the robot or while drawing, the test become invalid. Stability has been ensured by using a support specifically designed for this purpose. With the use of 3D mechanical drawing software and measuring the marker diameter, it has been possible to design the support. Once completed this operation, a 3D printer is used to create it.

Figure 101 - On the left, the 3D representation of the support.
On the right, the support realized with for the proper marker.

This stand is directly attached to the gripper, replacing the air blower and securely mounted to it using screws.

Figure 102 - On the left, the stand attached to the tool. On the right, the drawing operation.

The program executes a loop with 20 cycles, each corresponding to a point to be drawn. To better examine the dispersion of the cloud of points, each point will be placed along a beam of concentric circles spaced 1mm apart, with the target at the center. In this way, the repeatibility and the accuracy of the system can be graphically evaluated.



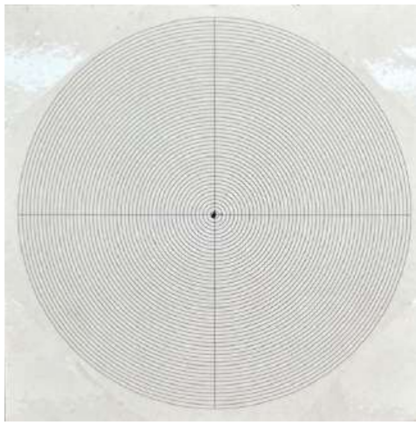


Figure 103 - On the left, a graphical explanation of accuracy and repeatibility [31]. On the right, the result obtained by this test.

As shown in the *Figure 103*, the cobot does not improve the uncertainties during the motion. Since the alignment error is known and the maximum error is estimated below 2mm, the points are drawn within the first two circles, as expected. With this result, the system demonstrates good accuracy and repeatability.

4.3 SIMULATION OF A REAL INDUSTRIAL APPLICATION

At the conclusion of this work, it may be useful to introduce a simulation of a real-world industrial application. So far, the error generated by the navigation of the MoMa in the space has only been simulated, now the real motion of the mobile robot is included in the test. In this way, it is possible to evaluate the overall behaviors and performances of the MoMa system in a realistic scenario. This approach allows for a more comprehensive analysis of both alignment and manipulation tasks, considering the combined effect of mobile platform positioning and robotic arm precision.

The test has been developed to simulate the real application of machine loading/unloading. Once the system has been programmed, the robot must approach a station defined in the space where to execute these operations.

A rigid metal support has been rigidly attached to a surface, so as to represent the robot workstation. At one end, the marker has been fixed: once the mobile platform reaches its position, the manipulator performs a movement toward the "*ShootPosition*", set at 250 mm from the marker. At this point, the vision system aligns the manipulator with respect to the marker using the method previously described. The program, after the alignment, acquires the TCP position and stores the coordinates in a new reference system. All movements toward the positions along the metal bar are executed with respect to this reference system.

The next sections describe the creation of a new map and the embedded base where to store objects. At the end, the program description is reported to clarify the functioning of the system.

4.3.1 SCENE RECONSTRUCTION

The first step, before introducing the movement of the mobile robot, is the creation of a map of the surrounding environment. The robot is manually guided through the space to scan the operative area using the laser radar. After completing this process, the map is displayed on the homepage of the Matrix interface. Here, some stations are included in the map: these serve as the positions among which the mobile robot moves to perform its tasks. In the middle, the figure of the robot is displayed, allowing to monitor the actual location of the robot in the space. The tracking of the movement between the stations is represented by connection lines.

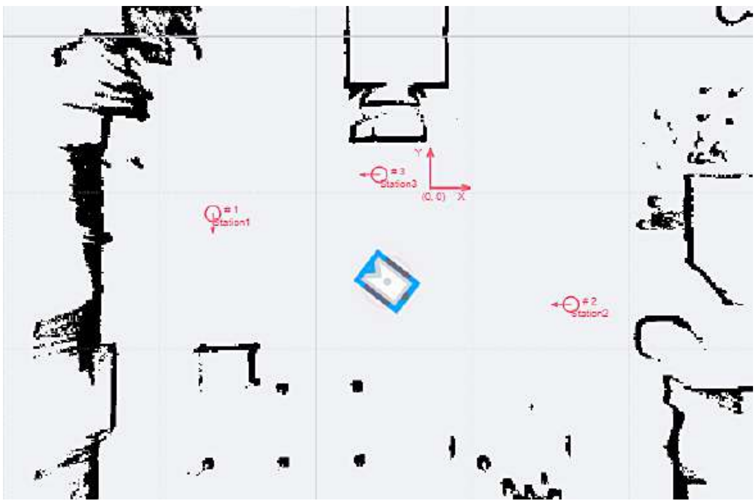


Figure 104 - Map of the environment.

For testing purposes, three stations have been defined and positioned at specific locations within the environment. The Station 3 is the one placed close to the plane where to execute the loading/unloading simulation. The Station 1 serves to guarantee an optimal exiting trajectory for the mobile robot. The Station 2 reproduces a drop-off point, in which some operations on board are executed.

The main program generates the movement toward the respective station by adjusting the values of the analog output connected to the Modbus register responsible for the autonomous navigation of the AMR. A wait instruction suspends the execution of the program until the mobile robot reaches the station. This is ensured by the analog input, which monitors the actual position of the robot relative to a specific station.

Figure 105 - AGV I/O instructions.

4.3.2 INITIAL SETTINGS

The considered operation mainly consists of a pick and place procedure for a sequence of objects. The system, as provided by the manufacturer, has a tray mounted on board. This support is used to store the objects during task execution. In the application it has been renamed as "*Pallet*". It is rigidly mounted above the integration structure, with pins that ensure always the same positioning relative to the robot base. The tray has a rectangular shape, with 10 slots built on it, constructed to hold the pieces.

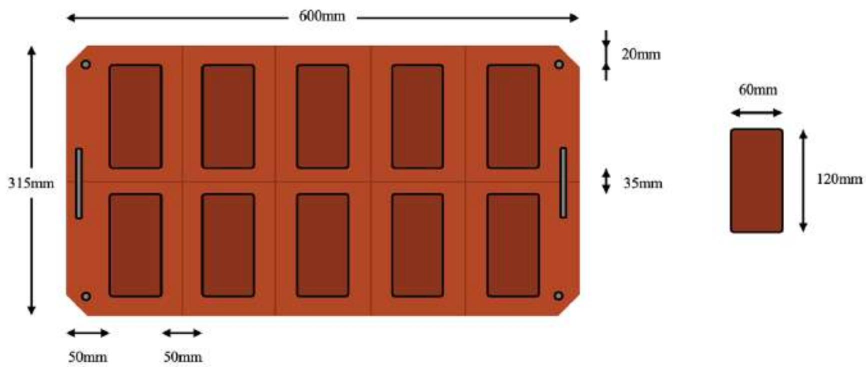


Figure 106 - Graphical representation of the on board tray.

The blocks have been designed using a 3D mechanical drawing software, according to the dimension of the slots, and subsequently printed with a 3D printer.

Figure 107 - On the left, the 3D representation of the block, while on the right its realization.

Since the "*Pallet*" can be considered as a rigid body, the positions of the slots remain unchanged. Therefore, it has been possible to define a reference frame, called "*PalletFrame*", and express the slot positions with respect to it. The definition of this new userframe has involved the creation of a tool for its configuration.

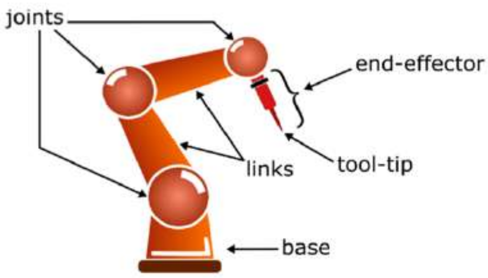


Figure 108 - Graphical representation of mounted tool-tip [32].

The starting point has been mounting a tip on the gripper, as previously described for the marker. The six-point calibration for the tool has been performed. This method has been necessary since the tip was not aligned along the same Z-axis direction as the flange. Another tip has been installed above the integration structure, acting as a fixed point in the space. The goal was to move the robot to six different positions, bringing the tip mounted on the tool in contact with the tip fixed on the structure, in six different configurations.

Figure 109 - One configuration of TCP creation.

The coordinates of these positions were recorded during the procedure, and, in the end, the calibration has been configured, obtaining the coordinates expressing the position of the "*ConfigTool*".

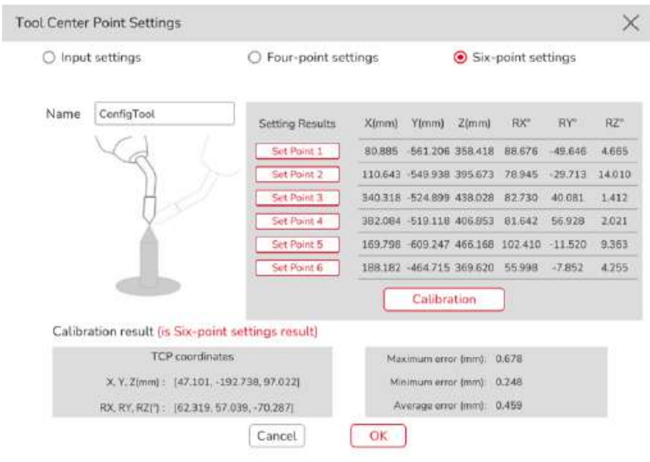


Figure 110 - TCP generation with six-point settings method.

ConfigTool	47.101	-192.738	97.022	62.319	57.039	-70.287

Figure 111 - TCP generation resulting coordinates.

To check the accuracy of the TCP generation, a simple test has been performed: the two tips have been aligned along the same vertical axis, with a distance of 1 mm between them. Then, by setting the motion around the TCP and executing some rotations, the tip endpoint had to remain fixed in space. At this point, it has been possible to configure the "*PalletFrame*" by touching three different points, one for the origin, one to identify the X-axis direction and the last one for the definition of the first quadrant of the XOY plane.

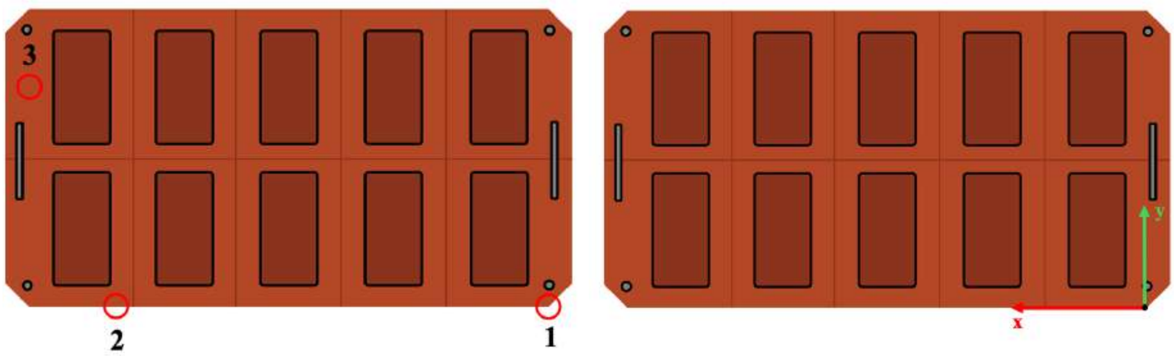


Figure 112 - On the left, the sequence of points to define the PalletFrame. On the right, the resulting coordinate system.

PalletFrame	-299.710	-274.069	96.469	179.714	-0.035	-1.064

Figure 113 - Coordinates result of PalletFrame generation.

The position of the closest slot to the origin of the "*PalletFrame*" has been estimated with a block closed in the gripper. The robot has been slowly moved until the block was placed in the slot: this point has been taken as a reference for the entire programming. Then, by following the same procedure, it was possible to define the position of the two adjacent blocks in X-Y directions. This operation allows the calculation of the variation along these directions, expressed as Δx and Δy. In this way, throughout the program, only the position of the first slot needs to be manually defined, the other positions are reached by adding the respective quantities.

As a preliminary test, a simple pick and place has been executed on the "*Pallet*", just moving the blocks from one row to the other. In this way it has been possible to validate the correctness of the measured slots points. This operation is integrated also in the overall procedure, executed when the MoMa is in the Station 2.

4.3.3 SYSTEM SETUP

Following the procedures described in previous sections and considering that the working conditions are different from those used before, it was necessary to execute the "*Debug mode*" to capture the marker placed in this different layout.

In this case, the position needs to be perfectly aligned with the workstation. For this purpose, using the same tip previously adopted and following a similar approach, a reference frame is also configured for the metal base. In this way, selecting a position with axis directions matching those of the metal base as a reference pose in "*Debug mode*" ensures that the robot is always aligned to the workspace, whenever the marker-based alignment is executed. This new userframe has been called "*LoadBase*".

LoadBase	-459.400	203.790	147.657	179.660	0.629	-91.022

Figure 114 - Coordinates result of LoadBase generation.

The first step concerns the definition of the new "*DebugPosition*", which is necessary for a new execution of the "*Debug mode*" settings. The coordinates of this pose are used to create a new userframe, "*AlignedUserframe*". This coordinate system serves to configure all the points of the workstation, computed through a procedure very similar to the one previously used for the slots of the on-board tray.

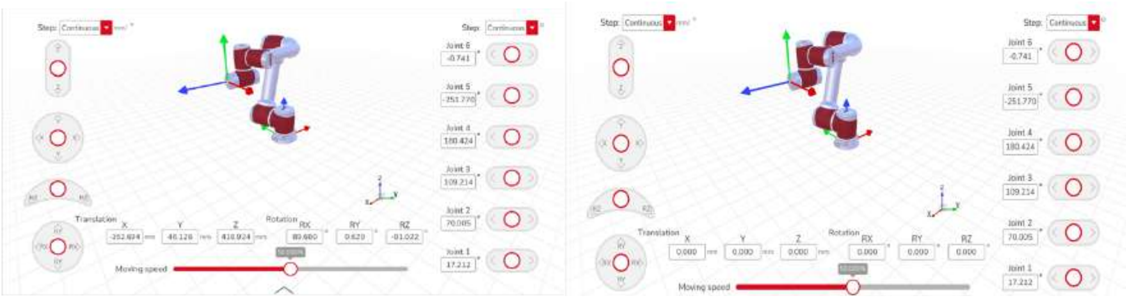


Figure 115 - Representations of the new "DebugPosition": on the left with coordinates of the TCP expressed with respect to the robot base, on the right the coordinates of the TCP expressed with respect to the new userframe.

AlignedUserframe	-352.694	46.126	416.924	89.660	0.629	-91.022

Figure 116 - New coordinate frame configured with the same coordinates as the new "DebugPosition".

These points represent the positions of the blocks on the metal base, computed with respect to this reference frame. Since the reference frame is updated after the alignment performed in "*Online mode*", the points defined with respect to it are also dynamically updated during execution. The new reference frame is stored in an array. This approach removes the constraint of using only *MoveL* instructions relative to that frame, as suggested by the manufacturer. After updating both the frame and the corresponding points, the cobot can reach these positions by normally using *MoveJ* instructions, which helps in singularities avoidance and allows greater flexibility.

Figure 117 - Programming instructions used to update the reference frame relative to which the motion is executed.

After this operation, it has been required to capture the reference image of the marker in the Lens X interface so to execute the localization. The camera parameters are not modified, exposure and gain are configured as in the previous tests to ensure full consistency with the results obtained from previous tests.

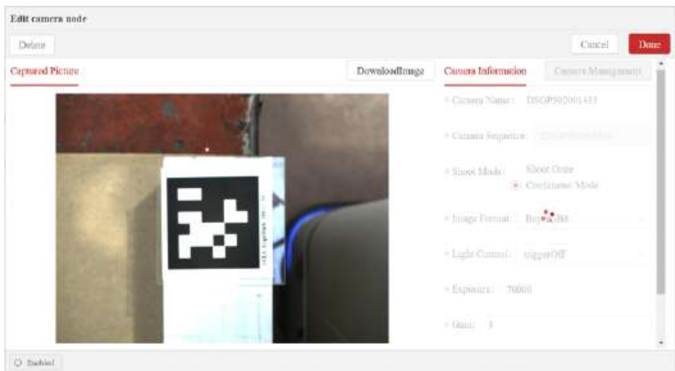


Figure 118 - JAKA Lens X real time scene capturing.

Once the localization has been successfully performed, a simple alignment test is required to validate the behavior of the robot in the new workspace.

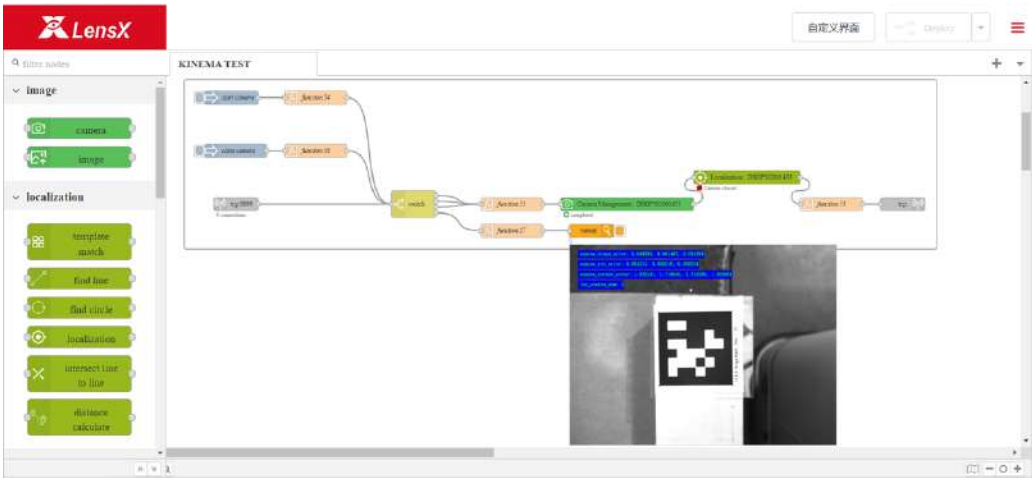


Figure 119 - Results of the alignment.

If all these operations produce the expected results, the set-up can be considered successfully completed. The system is then ready to perform the tasks assigned by the program.

4.3.4 OVERALL FUNCTIONING AND FINAL CONSIDERATIONS

The simulation developed aims at replicating a real industrial application. The blocks are used to represent any object that requires specific processing, such as drilling, bending, milling, welding or polishing. The workstation, reproduced by the metal base positioned at Station 3, simulates an external machine where objects must be placed to perform the desired process. This operation represents the core of the simulation, where the 2.5D alignment technique developed by JAKA and deeply analyzed in this work is applied.

For programming purposes, three new positions are defined in the space. One represents the rest position where the manipulator is located, while the AMR is moving, called "*Home*". The other two are waypoints: one is the home position on the pallet, where the robot aligns before moving onto the pallet, and the other one is relative to the metal base. They are called "*ReadyOnPallet*" and "*ReadyOnBase*", respectively.

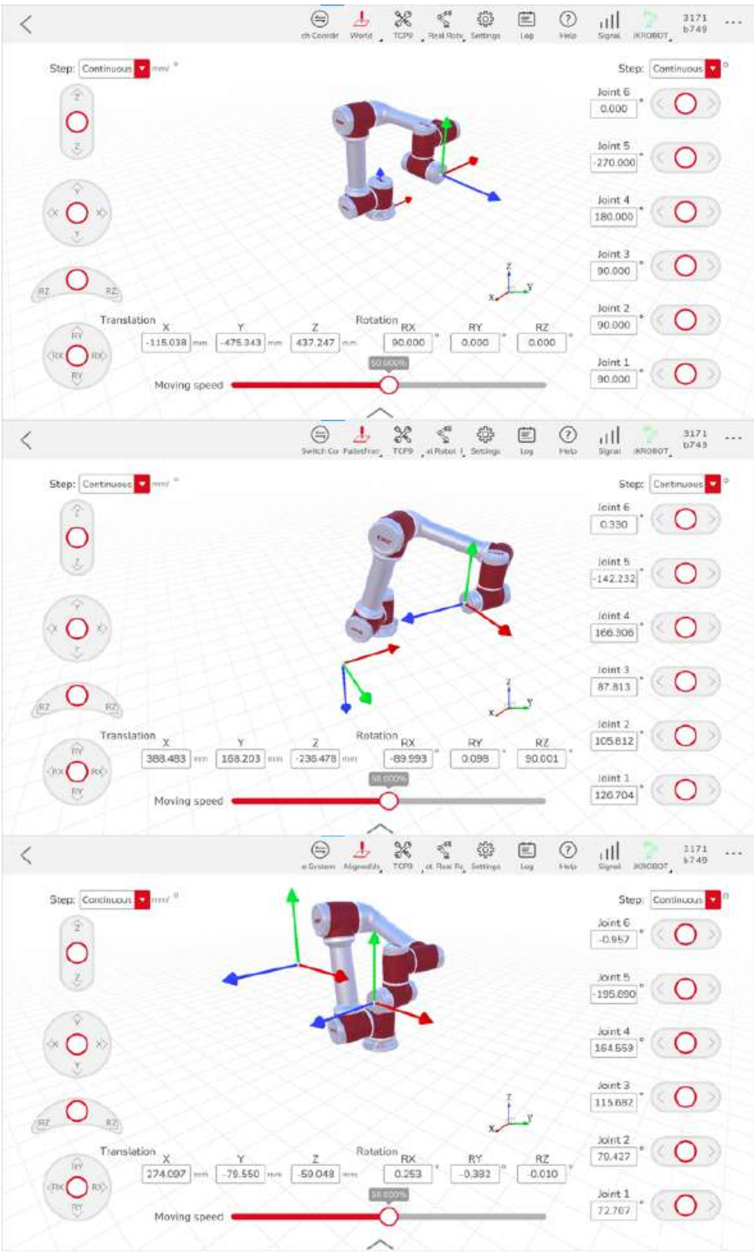


Figure 120 - On the top, the Home position, expressed with respect to the robot base. In the middle,the ReadyOnPallet position, expressed with respect to the PalletFrame. At the bottom the ReadyOnBase, expressed with respect to the AlignedUserframe.

The program starts by setting the instructions to move the MoMa to the Station 3, where the load/unload operations are executed when the robot arrives. During the execution of the program, the on-board light indicator is also controlled to signal the current operation. If the mobile robot is moving, the light is set to green. When the AMR stops at a station where the manipulator must perform a task, the light indicator turns to yellow.

Figure 121 - DynamicPickAndPlace robot program - initial settings.

At station 3, the cobot moves to the "*ShootPosition*", where all the alignment procedure is performed. The subsequent loop begins with the "*Load*" instruction, which places blocks on the base that is assumed to be empty at the start of the execution.

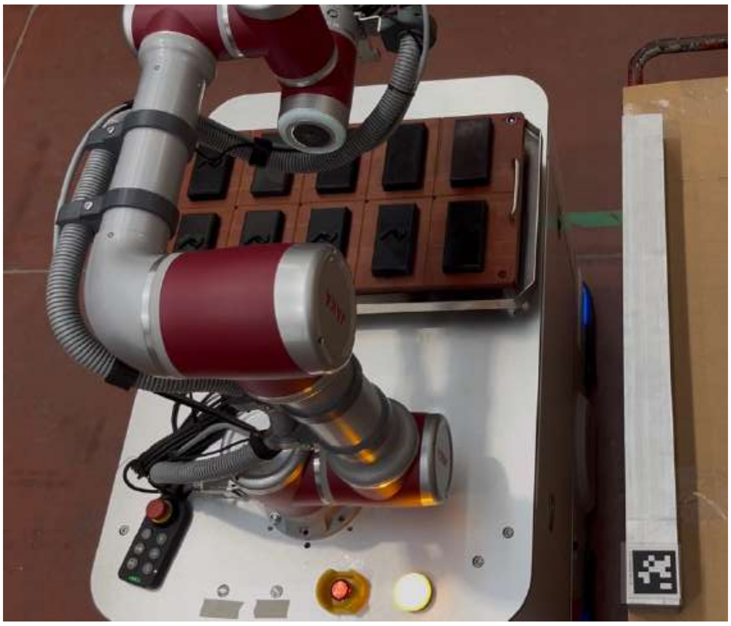


Figure 122 - First loading procedure of the cycle.

Since the unload operation always precedes the loading procedure, the movement to the "*Home*" position is only planned once the overall task is completed.

The loading and unloading procedures have been programmed following the same scheme. Each operation runs a loop, with each iteration involving one block and its respective positions. Namely, in the first loop of the loading procedure, the manipulator takes the block in the first position of the pallet and places it into the first position of the base.

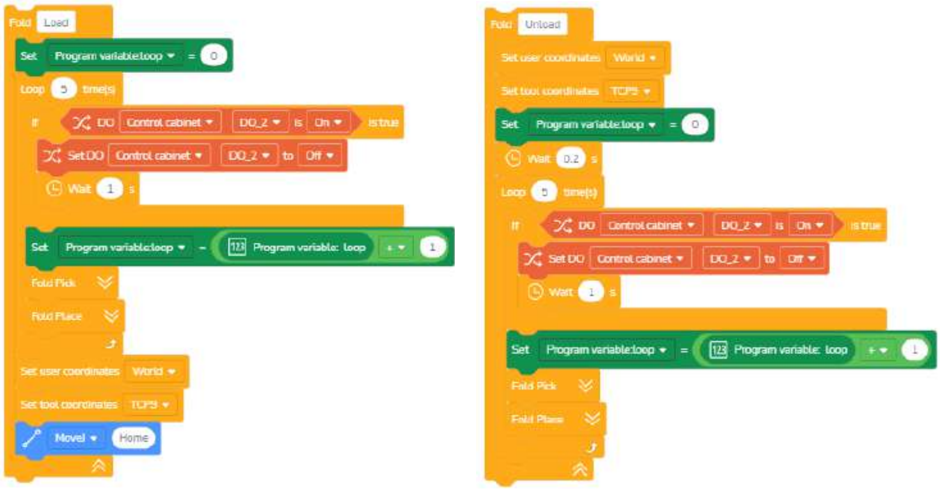


Figure 123 - Details of the DynamicPickAndPlace robot program.
On the left, the Load folder, on the right the Unload folder.

Figure 124 - On the left, the loading procedure. On the right, the unloading procedure.

Inside the folders "*Load*" and "*Unload*", containing the program necessary for these operations, there are two subfolders: "*Pick*" and "*Place*". Here, the main differences can be found, since the loading operation takes blocks from the pallet and places them on the base, while the unloading operation takes blocks from the base to store them on-board.

When the positions are reached, the MiniCab manages the opening/closing of the gripper setting the corresponding variables. In the *Figure 125*, "*Pick*" and "*Place*" folders of the "*Load*" operation are shown.

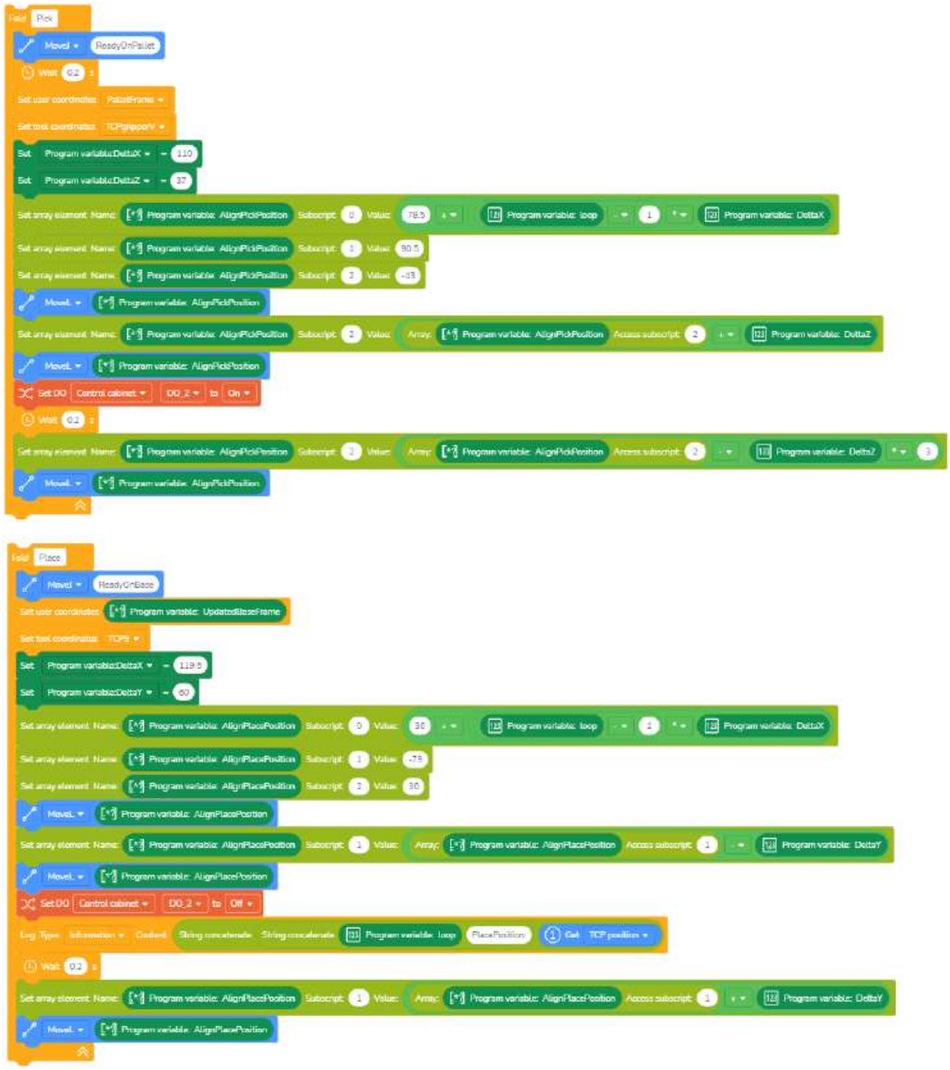


Figure 125 - On the top, the detailed commands inside the Pick folder. At the bottom, the detailed commands of the Place folder.

Once these operations are completed, the manipulator returns to the "*Home*" position and then the mobile robot transfers to station 1 where it simulates a standby, waiting for a few seconds. Afterward, the mobile robot proceeds to station 2 where the "*Exchange*" task is executed. This task involves a simple pick and place, shifting the blocks from the second row to the first one, reorganizing the pallet before the execution of the next tasks. This ensures that, during the next cycle, when the MoMa returns to Station 3, the second row will always be free for unloading the processed pieces, while the first row is ready for the loading. Moreover, station 2 is intended to simulate another loading/unloading station, similar to station 3. However, it was not possible to fully recreate the same working conditions as in station 3. For this reason, station 2 was used only for the pallet reorganization, to ensure the correct arrangement of the rows for the next processing cycle. This method guarantees an organized and repeatable flow of operations, keeping the two rows of the pallet always in the expected condition for the subsequent cycles.

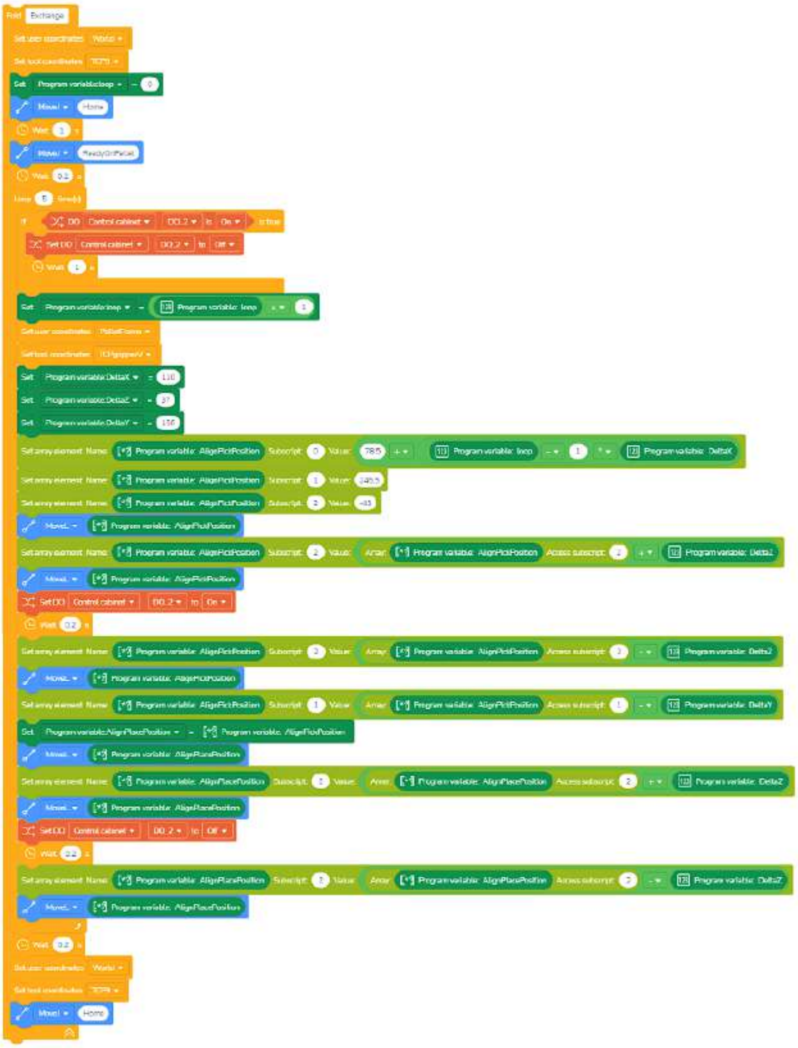


Figure 126 - Details of the Exhange operation of the DynamicPickAndPlace robot program.

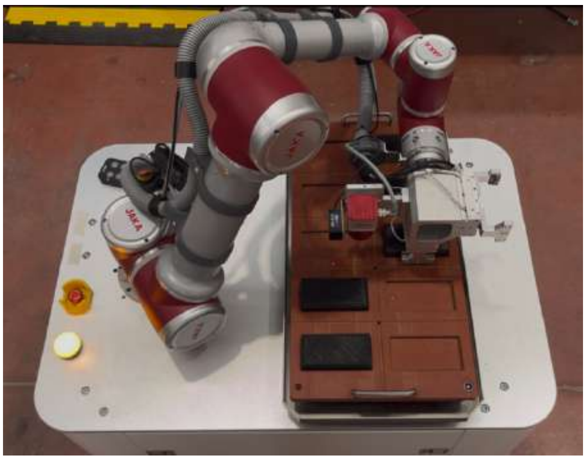


Figure 127 - The Exchange procedure.

At this point, the MoMa starts again its process with a new cycle of loading/unloading. The result of the alignment of the blocks on the base is shown in the *Figure 128*, where also the case in which the metal base changes its position is reported.

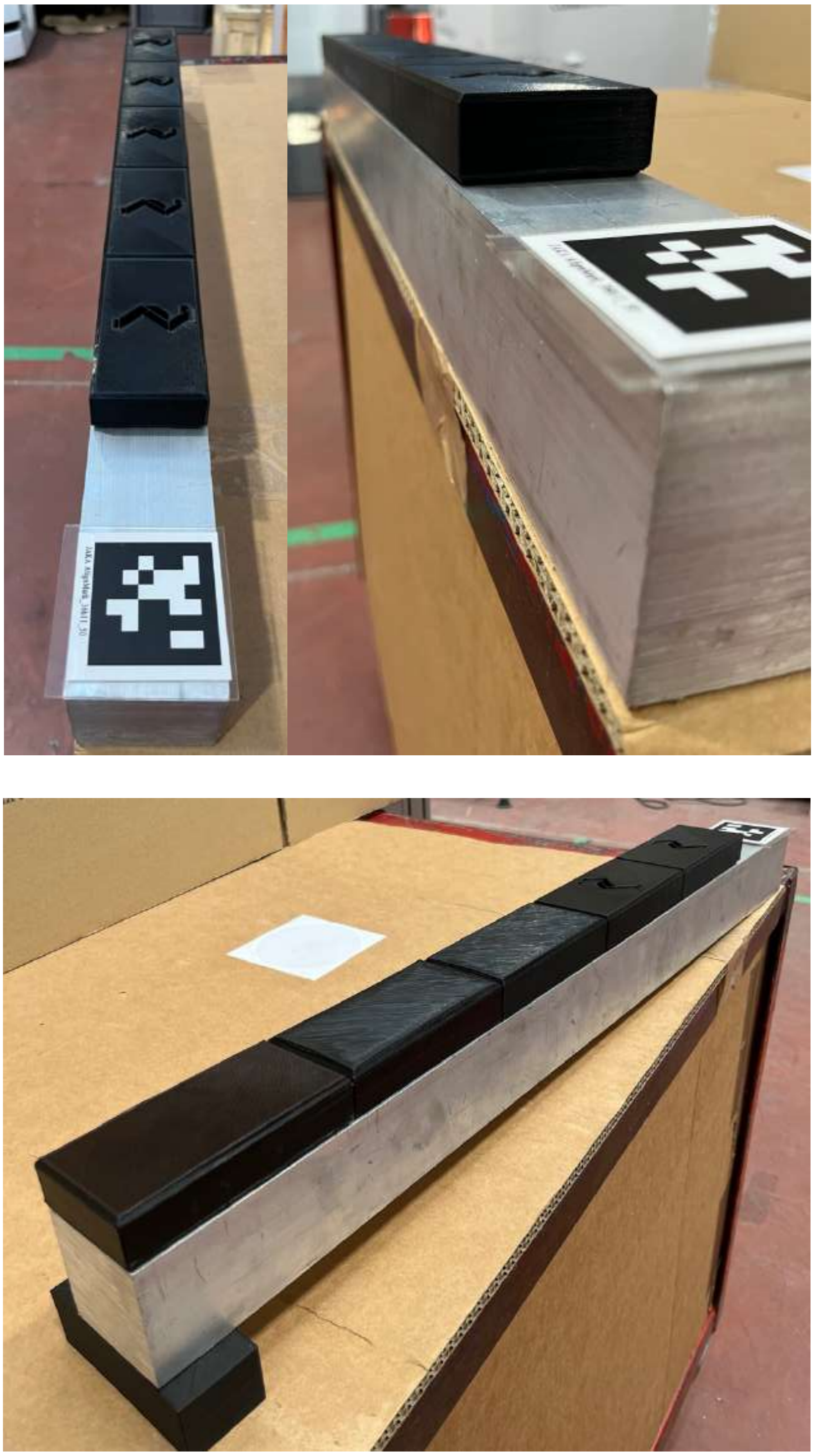


Figure 128 - On the top, the result of the loading operation without changes in the metal base. At the bottom, the result of the loading operation with slight rotations of the metal base about RY and RZ directions.

In conclusion, this final test has demonstrated that the system performs effectively, successfully completing the simulation of the intended industrial application. From the alignment procedure to the load/unload operations, each phase has been executed as expected, confirming the reliability and repeatability of the overall process.

It has been challenging to achieve high precision throughout the operations, particularly in tuning the camera parameters and the definition of the tool and the reference frame.

The effects of these uncertainties were particularly evident during the phase in which the blocks are placed on the bar. A rotational error of just a tenth of a degree, introduced during the alignment phase, combines with minor inaccuracies arising from the calculation of target points. It happened, during some placement operations, that while the first block was positioned precisely on the bar, the last block in the series touched the bar before releasing. A rotational misalignment of the cobot with respect to the marker-bar system, even as small as a few tenths of a degree, over the approximately 700 mm distance between the alignment point and the release point of the last block, results in a vertical displacement error of nearly 2 mm. For this reason, the manufacturer recommends a maximum working area distance of 500 mm from the marker to ensure optimal alignment accuracy.

These results suggest that, with improved alignment precision and more accurate manual calibrations, the proposed solution could be feasibly adopted in real industrial environments. The system has the potential to offer a flexible and automated approach to similar handling and processing tasks, enhancing efficiency and reliability in industrial workflows.

CONCLUSIONS

The contribution of this work lies in the thorough analysis of the functioning of a mobile manipulator prototype developed by JAKA. The combination of robotic arm manipulation capabilities with mobile robot mobility brings to the creation of mobile manipulators. These systems can significantly increase flexibility for various applications, however limitations in precision limit their expansion in certain fields. The manufacturer equipped the prototype with an embedded vision-based system with the purpose of mitigating the positioning error generated by the navigation of the mobile robot. This technique relies on a marker-based alignment combined with visual-servoing: they are implemented to introduce a feedback control loop that directly changes the manipulator position. In this way, it is possible to bring the cobot always in the same position with respect to the workspace, avoiding collisions or failure in the operations. The vision system is based on a 2D camera that uses a sequence of images taken from different perspectives to accurately estimate the height of the objects in its field of view. Once saved the reference position and captured the image of the marker, the camera will try to replicate the same view in future operations by adjusting the position of the manipulator. At this stage, the cobot will generate a motion toward a known point, moving relatively to the aligned position. Opting for a 2D camera instead of a 3D one will significantly reduce costs for the customer. This choice makes this technology more accessible and scalar for industrial applications.

The sequence of tests presented in this work highlight the strengths of this system. The combination of marker-alignment with visual-servoing generates a robust feedback mechanism that has effectively reduced the positioning errors, improving the accuracy of the overall system. While the alignment procedure gives positive outcomes, the limit of this system must be noticed. First of all, the region in which the cobot could operate is limited by the structure of the MoMa itself, since even the safe distance from obstacles must be considered when the

AMR approaches the working station. Then, the need of careful calibration of the parameters to obtain the optimal view of the marker could be challenging. The system requires working time and skilled staff to be adapted to customer necessities. Finally, the environmental conditions directly impact on the performances of the camera: the alignment still works even if the captured image is blur, but if the system will operate in a situation with substantial changes in lighting, the accuracy may introduce non negligible uncertainties. With further improvements in the alignment process, as can be the introduction of machine learning algorithms and trainings of the camera with a set of images reproducing different light conditions, it will represent a valuable tool for modern factories, offering adaptability to the surrounding environment and flexibility within operations. This system really presents a promising solution in various automation fields.

The introduction of a PLC may introduce further improvements to the prototype, replacing the MiniCab as the main controller of the system. Due to its deterministic behavior, the PLC would enable seamless integration of all the components, enhancing the robustness of the overall system. Its real-time capabilities allow synchronized operations between mobile robot, cobot and vision system, ensuring precise task execution and greater reliability. Nonetheless, the use of a PLC requires advanced programming skills to carefully manage the communication in the system. Moreover, this component could also introduce safety functionalities which were not addressed in this work. The safety management in industrial applications is a crucial aspect, especially when dealing with complex and composite robotic systems. There are two main components to implement safety functions: safety PLCs and safety relays, each one with its own characteristics.

Safety is a critical issue when human and robots collaborate in a shared workspace. When interacting with human workers, new hazards arise that are currently not covered in any safety standard. The new technologies will never be widely accepted by the industry if they are not safe, in relation to both humans and surrounding environment. Robots must be equipped with adequate sensors connected to the safety module that perform the desired countermeasures, according to safety standards. In the case of MoMa, specific criteria must be applied. The manipulator can be considered as a load and the risk assessment for the total hazard may be evaluated only considering the mobile robot. The cobot must be contracted in a stowed rest position when the mobile platform is moving. The definition of this safe position is a critical aspect, for which it is better to refer to robotic standards. The whole application must be evaluated according to part 2 of the EN ISO 10218 standard, which is dealing with the integration of industrial applications. There are currently no fully-compliant standards, guidelines or design proposal for MoMa, new approaches and safety models are highly needed.

Nowadays, MoMa are semi-autonomous and pre-programmed systems. However, the industry environments may be variable and require adaptability to different scenarios. Until now, the system has not reached a sophisticated level of intelligence. The future of MoMa will be made by fully autonomous systems, allowing the system to carry out entire tasks on its own. This may increase the maturity of systems in terms of the fourth industrial revolution, with on board intelligence to be independently able to spot mistakes, evaluate risks and handle different situations. AI has the potential to advance this topic.

Most navigation methods involve maps, but in a situation where environmental information is limited, the inconsistency of the environment makes these methods inadequate. Given the rapid advancement in deep learning and computer vision, low-cost vision sensors have found increasingly applications in vision-based robot navigation. Recent advancements in artificial intelligence suggest that learning-based methods will emerge as the major approach in future navigation systems. Deep Reinforcement Learning (DRL) combines the strengths of deep learning and reinforcement learning, offering potential for continued progress in autonomous navigation as the field advances. Specifically, can be adopted a reinforcement learning framework based on deep Q networks (DQN), directly utilizing the image graph provided by the RGB-D camera as a representation of state space. Experimental results have shown that even if the noise of RGB image is very large, the robot can successfully navigate around the environment and reach the target destination while achieving effective obstacle avoidance [14].

The current trend in collaborative robotics is concentrated on developing flexible systems that enable safe and cooperative relations between humans for a variety of tasks. There is an increasing interest in the area of collaborative robotics to make it possible for humans to teach robots different types of skills. Allowing robots to learn from human demonstration means that they first need to be able to learn and recognize meaningful actions [19].

Additionally, the robot needs to perceive the environment in order to adapt to changes in the workspace, such as shifts in the object of interest during the collaborative task or variations in the position of the user's hand [23].

Some concepts are proposed to expand the description of the technological process of human and cobot collaborative work: Interactive Motion Control of a robot-manipulator allows a person to directly controls individual actions of a robot-manipulator using gestures, voice

commands or a special interface. Predictive Human-Robot Interaction assumes that the robot can predict the operator's actions based on sensor data, analyzing the operator's movements and position. Interactive Learning System allows the robot to learn directly from the operator, remembering actions and improving its own algorithms for further tasks.
These new concepts will help in creating efficient technological processes using collaborative manipulator robots, promoting deeper interaction between humans and robotic systems. The research and development of new strategies will bring us to industry 5.0, increasing efficiency, flexibility and safety of production processes [33].

Intelligent feedback, adaptive tasks, and multimodal integration foster more natural and harmonious cooperation between humans and robots. This collaborative approach is a key success factor in the new era of robotics, where personalizing production processes and focusing on human needs are increasingly important. However, the challenge of balancing automation with human involvement in industry persists. Future advancements in AI may provide insights and solutions to address this issue.

BIBLIOGRAPHY

1. R. Galin and R. Meshcheryakov, “Automation and robotics in the context of Industry 4.0: the shift to collaborative robots,” *IOP Conference Series Materials Science and Engineering*, vol. 537, no. 3, p. 032073, May 2019, doi: 10.1088/1757-899x/537/3/032073.
2. Z. Feng, G. Hu, Y. Sun, and J. Soon, “An overview of collaborative robotic manipulation in multi-robot systems,” *Annual Reviews in Control*, vol. 49, pp. 113–127, Jan. 2020, doi: 10.1016/j.arcontrol.2020.02.002.
3. J. L. Outón, I. Villaverde, H. Herrero, U. Esnaola, and B. Sierra, “Innovative mobile manipulator solution for modern flexible manufacturing processes,” *Sensors*, vol. 19, no. 24, p. 5414, Dec. 2019, doi: 10.3390/s19245414.
4. N. Ghodsian, K. Benfriha, A. Olabi, V. Gopinath, and A. Arnou, “Mobile Manipulators in Industry 4.0: A Review of Developments for Industrial applications,” *Sensors*, vol. 23, no. 19, p. 8026, Sep. 2023, doi: 10.3390/s23198026.
5. J. L. Outón *et al.*, “A Real Application of an Autonomous Industrial Mobile Manipulator within Industrial Context,” *Electronics*, vol. 10, no. 11, p. 1276, May 2021, doi: 10.3390/electronics10111276.
6. M. A. Roa *et al.*, “Mobile Manipulation Hackathon: Moving into Real World Applications,” *IEEE Robotics & Automation Magazine*, vol. 28, no. 2, pp. 112–124, Mar. 2021, doi: 10.1109/mra.2021.3061951.
7. Yang, M. *et al.* (2019) ‘Collaborative Mobile Industrial Manipulator: A review of system architecture and applications’, *2019 25th International Conference on Automation and Computing (ICAC)*, pp. 1–6. doi:10.23919/iconac.2019.8895183.
8. P. Stibinger *et al.*, “Mobile Manipulator for autonomous localization, grasping and precise placement of construction material in a Semi-Structured environment,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2595–2602, Feb. 2021, doi: 10.1109/lra.2021.3061377.
9. Y. You *et al.*, “Design and Implementation of Mobile Manipulator System,” *9th IEEE International Conference on CYBER Technology in Automation, Control, and Intelligent Systems*, Jul. 2019, doi: 10.1109/cyber46603.2019.9066594.
10. M. Hvilshøj, S. Bøgh, O. S. Nielsen, and O. Madsen, “Autonomous industrial mobile manipulation (AIMM): past, present and future,” *Industrial Robot the International Journal of Robotics Research and Application*, vol. 39, no. 2, pp. 120–135, Mar. 2012, doi: 10.1108/01439911211201582.
11. M. Mihelj *et al.*, *Robotics*. Springer, 2018.
12. Markis, Alexandra & Papa, Maximilian & Kaselautzke, David & Rathmair, Michael & Sattinger, Vinzenz & Brandstötter, Mathias. (2019). Safety of Mobile Robot Systems in Industrial Applications. 10.3217/978-3-85125-663-5-00.

13. Balachandran, Adithya & S, Anil & Sreedharan, Pramod. (2022). Autonomous Navigation of an AMR using Deep Reinforcement Learning in a Warehouse Environment. 1-5. 10.1109/MysuruCon55714.2022.9971804.
14. Yin, Yan & Chen, Zhiyu & Liu, Gang & Yin, Jiasong & Guo, Jianwei. (2024). Autonomous navigation of mobile robots in unknown environments using off-policy reinforcement learning with curriculum learning. Expert Systems with Applications. 247. 123202. 10.1016/j.eswa.2024.123202.
15. F. Gul, W. Rahiman, and S. S. N. Alhady, "A comprehensive study for robot navigation techniques," *Cogent Engineering*, vol. 6, no. 1, Jan. 2019, doi: 10.1080/23311916.2019.1632046.
16. Diginsa, Muhammad & Shafie, Noraimi & Yusuf, Nazir. (2023). Review: Issues and Challenges of Simultaneous Localization and Mapping (SLAM) Technology in Autonomous Robot. International Journal of Innovative Computing. 13. 59-63. 10.11113/ijic.v13n2.408.
17. Munoz Ceballos, Nelson & Valencia Velasquez, Jaime & Alvarez, Alessa. (2013). Simulation and Assessment Educational Framework for Mobile Robot Algorithms. Journal of Automation, Mobile Robotics & Intelligent Systems. 8. 10.14313/JAMRIS_1-2014/7.
18. Weidemann, Carlo & Mandischer, Nils & Kerkom, Frederick & Corves, Burkhard & Hüsing, Mathias & Kraus, Thomas & Garus, Cyryl. (2023). Literature Review on Recent Trends and Perspectives of Collaborative Robotics in Work 4.0. Robotics. 12. 84. 10.3390/robotics12030084.
19. Patil, Swapnil & Vasu, Vishwa & Srinadh, K.. (2023). Advances and perspectives in collaborative robotics: a review of key technologies and emerging trends. Discover Mechanical Engineering. 2. 10.1007/s44245-023-00021-8.
20. Franklin, Carole & Dominguez, Elena & Fryman, Jeff & Lewandowski, Mark. (2020). Collaborative robotics: New era of human–robot cooperation in the workplace. Journal of Safety Research. 74. 10.1016/j.jsr.2020.06.013.
21. Kragic, Danica & Gustafson, Joakim & Karaoguz, Hakan & Jensfelt, Patric & Krug, Robert. (2018). Interactive, Collaborative Robots: Challenges and Opportunities. 18-25. 10.24963/ijcai.2018/3.
22. Safeea, Mohammad & Neto, Pedro & Béarée, Richard. (2019). On-line collision avoidance for collaborative robot manipulators by adjusting off-line generated paths: An industrial use case. 10.48550/arXiv.1909.05159.
23. Semeraro, Francesco & Griffiths, Alexander & Cangelosi, Angelo. (2023). Human–robot collaboration and machine learning: A systematic review of recent research. Robotics and Computer-Integrated Manufacturing. 79. 102432. 10.1016/j.rcim.2022.102432.
24. Roth, Zvi & Zhuang, Hanqi & Wang, Luke. (1993). Simultaneous Calibration of a Robot and a Hand-Mounted Camera.. 149-154. 10.1109/ROBOT.1993.292139.
25. Corke, Peter. (2017). Robotics, Vision and Control. 10.1007/978-3-319-54413-7.
26. Siciliano, Bruno & Sciavicco, Lorenzo & Luigi, Villani & Oriolo, Giuseppe. (2011). Robotics: Modelling, Planning and Control. 10.1007/978-1-84628-642-1.
27. Tsai, Roger & Lenz, Reimar. (1988). Real time versatile robotics hand/eye calibration using 3D machine vision. 554 - 561 vol.1. 10.1109/ROBOT.1988.12110.
28. Wells, Gordon & Venaille, Christophe & Torras, Carme. (1996). Vision-based robot positioning using neural networks. Image and Vision Computing. 14. 715-732. 10.1016/0262-8856(96)89022-6.

29. Korayem, M. & Ghariblu, Hashem. (2004). Analysis of wheeled mobile flexible manipulator dynamic motions with maximum load carrying capacities. Robotics and Autonomous Systems. 48. 63-76. 10.1016/j.robot.2004.07.010.
30. Slamani, Mohamed & Nubiola, Albert & Bonev, Ilian. (2012). Assessment of the positioning performance of an industrial robot. Industrial Robot: An International Journal. 39. 57-68. 10.1108/01439911211192501.
31. Shiakolas, Panos & Conrad, Kevin & Yih, T.C.. (2002). On the Accuracy, Repeatability, and Degree of Influence of Kinematics Parameters for Industrial Robots. International journal of modelling and simulation. 22. 245-254. 10.1080/02286203.2002.11442246.
32. Perez, Luis & Rodríguez, Íñigo & Rodríguez, Nuria & Usamentiaga, Rubén & García, Daniel. (2016). Robot Guidance Using Machine Vision Techniques in Industrial Environments: A Comparative Review. Sensors. 16. 335. 10.3390/s16030335.
33. Yevsieiev, Vladyslav & Abu Jassar, Amer & Maksymova, Svitlana & Gurin, Dmytro. (2024). Human Operator Identification in a Collaborative Robot Workspace within the Industry 5.0 Concept. 4. 95-105.