UCLouvain

epl

École polytechnique de Louvain

Design and Control of a Delta Robot for Pedagogical Purpose

Authors: **Victor De Maegd**[1], **Paolo Mollica**[2]
Supervisors: **Paul Fisette, Renaud Ronsse**
Readers: **Benoît Herman, Emile Moreau**
Academic year 2023–2024
1. Master [120] in Electro-mechanical Engineering
2. Master [120] in Mechanical Engineering

Abstract

The objective of this master's thesis is to enhance the control and performance of a Delta robot designed for educational purposes. This robot will be showcased at the "Printemps des Sciences" event in Louvain-la-Neuve to demonstrate its capabilities to a young audience. The robot must execute original and fun tasks that highlight its full range of abilities.

This thesis represents the second iteration of the project, with a focus on measuring the robot's accuracy, repeatability, and flexibility. Additionally, it addresses motion control and the design of a gripper system. The ultimate goal is to develop a Delta robot capable of performing educational demonstrations with high precision.

A pneumatic gripper was designed and optimized for tasks such as pick-and-place operations, enhancing the robot's ability to perform engaging demonstrations. Furthermore, a user-friendly graphical interface was developed using ElectronJS, simplifying robot control for users with minimal programming experience. This interface allows easy execution of pre-defined trajectories, improving the robot's usability in educational settings.

The robot's performance was rigorously evaluated in terms of pose accuracy, repeatability, and flexibility, in compliance with ISO standards. Results show the robot achieves a pose accuracy of less than $1[mm]$ and repeatability within $\pm 0.5[mm]$, making it well-suited for illustrating key robotics concepts.

The thesis also includes detailed documentation of the robot's programming, providing sufficient information for users to modify the code and add new trajectories. The code used an inverse kinematic model to computes the angles of the motor based on the trajectory and motion laws.

In conclusion, this thesis successfully presents an educational Delta robot, combining gripper design and its integration, and intuitive programming to allow captivating demonstrations.

Acknowledgement

This thesis represents the culmination of extensive work and dedication, and it would not have been possible without the support and contributions of several key individuals. We would like to take this opportunity to express our deepest gratitude to those who played an integral role in helping us bring this project to fruition.

First and foremost, we would like to thank our supervisors, Mr. Paul Fisette and Mr. Renaud Ronsse, for their trust in us despite our clear communication issues and for their helpful criticism on the thesis's development.

We are grateful to Emile Moreau for his weekly oversight of the thesis and the numerous guidance he provided us with this year.

Our sincere thanks go to Simon De Jaeger for his daily assistance and technical support. His involvement extended far beyond routine tasks, including the setup of precision testing equipment, the critical fixation of the Delta robot, and the the printing of various 3D parts. Simon's attention to detail and problem-solving skills were precious in overcoming the technical challenges we faced.

We also wish to acknowledge the significant contributions of Alex Bertholet, who generously lent us his lasers and helped us installing the LEMSC-tech. In a similar vein, we thank Antoine Bietlot for his valuable support with laser data acquisition.

Our deep appreciation extends to Thierry Daras for his help regarding the implementation of the pneumatic system. Additionally, we are deeply grateful to Frank Hesbois for his assistance with the high-speed camera, which enabled us to capture the precise data that was crucial to our analysis.

We would also like to thank Benoît Herman in advance for the time and effort he will invest in reviewing this thesis.

We would be remiss not to mention the unwavering support of our families. Their persistent encouragement, both moral and logistical, provided us with the foundation needed to persevere through the challenges of this project. Their belief in us has been a source of strength and motivation.

Lastly, we want to acknowledge that we used ChatGPT and DeepL during the writing process: ChatGPT assisted us in refining the text structure and phrasing, while DeepL helped with translating certain sentences from French to English.

Contents

Introduction

The Delta robot is a specialized parallel robot known for its unique design, featuring three arms connected to universal joints at the base. This configuration allows the robot to achieve exceptional speed and precision, making it ideal for high-speed pick-and-place operations, assembly, and packaging in industries such as electronics, pharmaceuticals, and food processing. Developed in the 1980s by Professor Reymond Clavel at the École Polytechnique Fédérale de Lausanne in Switzerland, the Delta robot's innovative structure minimizes the inertia of its moving parts, enabling agile and accurate operation. It has since become a standard in automation, celebrated for its efficiency and versatility in performing repetitive tasks with precision.

Context

UCLouvain began its mission to develop its own Delta robot last year, with two final-year engineering students from l'École polytechnique de Louvain undertaking the project for their thesis. Starting from scratch, they fully designed and assembled the robot. Their project included thorough modeling of the robot's geometric, kinematic, elastic, and dynamic behavior to optimize its performance. The thesis also covered the detailed design of mechanical parts, the selection of motors and electronics, and the validation of the robot through various trajectories. Additionally, it provided extensive documentation to support future iterations and improvements.

Project Continuation

This master's thesis represents the second iteration of the Delta robot developed at EPL, aiming to deliver a fully functional demonstration robot for pedagogical purposes.

The thesis begins by outlining metrics used in the industrial sector to test the performance of industrial robots. It then reviews the specifications set in the

previous year's thesis, highlighting which goals were achieved and which required adjustments or additions. One notable addition is the design and implementation of a gripper for the robot's end-effector, which is thoroughly explained in the following sections.

Next, the thesis details the development of a graphical user interface (GUI) for the robot, explaining the purpose and benefits of this feature.

Based on the state of the art, the performance testing of the robot is then carefully analyzed, providing precise measurements of the Delta robot's accuracy and consistency. Additionally, an experimental method to assess the flexibility of the robot's arms is proposed and executed.

Finally, the robot's demonstration capabilities are described, along with their pedagogical purposes.

Chapter 1

State of the art - Performance testing in industrial application

The term “robot” was derived from English translation of a Czech play in 1920. However, it took another 40 years before industrialists could benefit from the modern technology of robotic manipulators [1]. Advancements in the domain of robotics have redefined automation, which now involves use of mechatronics systems in the operation and control of production. Computer technology and modern electronics make it possible to realize efficient and economical solutions for industrial processes. Robotics now is a form of industrial automation which offers benefits in terms of programming feature and the resulting adaptability of the equipment. This facilitates economical manufacturing of many different and unique products in small batches [2]. In this pursuit of economic efficiency, a key question emerges: how can performance metrics be used to determine which robot outperforms another?

1.1 Context

As explained earlier, this work is a continuation of a master’s thesis [3] in which the students designed and built a Delta robot. This raises the question of the performance of the resulting robot. Indeed, one might want to evaluate its various specifications such as workspace size, position accuracy, speed, position repeatability, accelerations to see for what specific applications one might use this robot.

In this particular case, where the robot serves primarily an educational purpose, its performance may not be of the utmost importance. However, evaluating the performance of a robot can still be valuable, as it helps to better understand the

robot's capabilities and limitations. Moreover, it ensures that the robot functions optimally for its intended educational goals. For other cases, the evaluation of different metrics is crucial. For example, when purchasing a robot (e.g., to work on a production line), we may want to be able to easily compare different robots to find the one best suited to our specific application. This process bridges the gap between academic learning and industrial application, preparing students for the demands of modern robotics in both research and professional settings.

Research on general robot performance dates back to the 1980s, a time when the field was less dispersed than it is now. This is explained by the fact that there were fewer types of robotic systems available on the market, most of which were industrial robotic arms [4]. However, according to [5], the demand for industrial robots is growing quickly. The market is changing and will not only include big companies anymore. It will also reach small and medium-sized manufacturers that lack the necessary funding to hire experts in robotics. Therefore, it is essential to establish clear guidelines for selecting the right robotic components for different applications, especially to assist those who may not be experts in the field. For users to be able to choose and assess only the specific aspects of a robot system that are significant to them, a set of performance metrics for each component needs to be available. Therefore, defining general and clear performance metrics might be very useful.

Numerous tests and performance measures have been proposed to assess robot capabilities. These have been developed both informally by academics who wanted to assess their systems and formally through standards. However, they are often tailored for specific kinds of robots or applications rather than aiming for a unified framework [4]. Furthermore, the formal performance standards for industrial robots (ANSI/RIA R15.05 [6, 7, 8], ISO 9283 [9]) have not changed since a long time (developed in 1998 and not updated since then for ISO-9283) and only address a small number of performance-related topics. Conversely, a concentrated effort has been made to create performance requirements for mobile robots [10]. A few of them might also be valuable in industrial applications.

The following sections will discuss and compare existing performance metrics, highlighting how these metrics are valuable both in the design phase of a robot and in assessing its capabilities.

1.2 Classification of performance metrics

To better understand the complexities of robot performance, it is essential to categorize these metrics based on specific criteria, which will allow for a more

structured analysis and comparison.

1.2.1 Performance metrics/Indices definition

Before classifying and comparing *performance metrics* for industrial robots, it is crucial to define what performance metrics are. The term *performance* refers both to the execution of a task and the quality of that execution [11]. In the field of industrial robotics, performance metrics are used to evaluate and compare the capabilities of robots. These metrics, which are often numerical, measure how accurately, effectively, and efficiently a robot can perform specific tasks [12].

These quantitative metrics have a lot of benefits in engineering, indeed as explained by Chang in [13] : «[...], a quantitative measure provides one with a rational basis upon which one can, without having to rely on experience and intuition alone, analyze, design, and control the systems». These performance indices are crucial for evaluating a system's performance, a primary focus of this master thesis, and for guiding its design to achieve targeted performance levels. Optimizing these metrics allows for precise control and fine-tuning of the system to meet desired outcomes, with ongoing adjustments as necessary.

For these metrics to be effective, they must be meaningful and consistent. General performance indices are useful for comparing a wide range of systems, while specific metrics are needed to capture particular aspects of robot behavior. To balance these needs, metrics are classified based on their dependence on factors such as pose/scope, task/application, scale/unit, physics, and safety [4, 12, 14].

In the following sections, the terms *indices* and *metrics* will be used interchangeably.

1.2.2 Pose-dependent metrics

This first category includes metrics that vary based on the robot's current configuration or position. These indices are classified as either local or global.

- **Local performance indices**: These metrics indicate performance for a specific configuration/posture of the robot, such as end-effector position or joint angles in the workspace. They are pose-dependent and vary point by point across different configurations. While they prevent fast comparison to other robots, they have higher complexity and lack a single value to describe the entire system, they can, on the other hand, be used to detect kinematic singularities. These indices are very useful in manipulator control applications. Most Jacobian-based performance metrics, such as the manipulability index and condition number (which are detailed further), are local indices because

the Jacobian varies with the robot's configuration at different points in the workspace [15, 12].

- **Global performance indices**: On the contrary, global metrics do not depend on the robot's configuration. They provide a measure of the robot's performance across its entire workspace. They are necessary to compare performances on the same application of two different manipulators/robots. They can also be extensions of local metrics, calculated by averaging or taking extreme values of local indices across the workspace [16]. However, global ones may overlook variations in performance. To address this, the worst value of a local index is often used as the global metric, reflecting the lowest performance across the workspace. Be aware that a high local performance index does not necessarily produce a high global performance index, and the opposite is also true [17]. For instance, some robot manipulators described in the following study [18] have high local index values, but the global performance index of these manipulators is not high compared to their local index values.

1.2.3 Physics dependencies

Another category used in the literature to classify performance metrics is based on the type of physical behavior they assess.

- **Kinematic indices**: These metrics depend on the robot's geometry/structure [19] and topology, and are therefore based on the Jacobian (a notion explained in detail a bit further Section 1.3.2). An example of kinematic metrics includes accuracy, repeatability, speed, and workspace (we usually find their values on robots data-sheets).
- **Dynamic indices**: These metrics assess the robot's performance in terms of the loads it can handle (e.g. payload capacity and power consumption). If the system's inertial effects are negligible, static metrics can be used instead to simplify the problem.

This master thesis focus more on mechanical performance (kinematics and dynamics), however one could use metrics from other physics field (e.g. electromechanical or thermal as in [20]).

1.2.4 Task-based performance metrics

Industrial robots can perform a wide range of tasks, including material handling, assembly, welding, and more. Therefore, specific performance metrics are needed to evaluate their capabilities for these different tasks. These metrics can be classified as either intrinsic or extrinsic.

- **Extrinsic indices**: When the robot's use case involves a clearly defined set of tasks, extrinsic metrics become more relevant because they assess the robot's performance under specific, task-oriented requirements. Typically, extrinsic indices are preferred for specialized operations that demand performance near the system's technological limits. Examples include manipulators for tasks such as precision surgical procedures, high-speed sorting in logistics, and intricate assembly processes in electronics manufacturing.
- **Intrinsic indices**: these indices are unrelated to the robot's application or task. Strong performance on intrinsic metrics indicates a robot's general capability to perform well across various tasks, making these metrics ideal for robots intended for general or flexible use. Therefore, intrinsic indices are favored in open-ended applications. Some examples of intrinsic metrics are dexterity index, manipulability, and condition number [12].

1.2.5 How can we quantify those performance metrics?

Now, that different classifications have been made, the goal is still to convert the robot's behavior into values that are simple to understand. To do this, performance metrics can be expressed as absolute or relative/dimensionless values.

- **Relative metrics**: This kind of indices, which compare performance to a reference value, are dimensionless and facilitate quick comparisons. These metrics are particularly useful for evaluating two different robotic structures, or two versions of the same robotic structure [14]. Although selecting a meaningful reference can be challenging, relative metrics are preferred for their ability to indicate success or failure margins. For example, if one wants to evaluate the payload capacity utilization of an industrial robot, a relative metric can be used to determine how close the robot is operating to its maximum payload capacity. If the robot is designed to handle a maximum load of 100 kg but regularly operates at 90 kg, the relative metric of load capacity utilization would be 90%. It represents a 10% failure margin, which indicates the proximity to potential overload conditions.
- **Absolute metrics**: These metrics, measured in units like kilograms for payload or meters per second for speed, are intuitive and straightforward but can be less useful without context and difficult to compare across different systems. For example, knowing a robot has a 4 kg payload is clear but doesn't help compare with another robot's percentage capacity.

In the previous sections, various categories of performance metrics have been introduced; however, other classifications also exist, such as Safety-based performance metrics [14], Energy efficiency metrics [21], and Environmental impact metrics.

1.3 Performance indices

1.3.1 Common metrics

This section introduces and categorizes various performance metrics as described in the previous section. The metrics listed in Table 1.1, come from both robots datasheets and standard organizations [22]. These indices are used for a wide range of robots, manufacturers, and markets. They are also frequently used in research to describe and optimize robot designs [12]. However, because of their generality, they are better suited to preliminary design selection than fine-tuning robot performances (where more specific metrics will be used).

Accuracy and repeatability are classified differently by commercial datasheets and standard organizations. In commercial datasheets, they are typically listed as global intrinsic metrics, representing the overall performance across the entire workspace of the robot. On the other hand, standard organizations classify these metrics as local extrinsic metrics, focusing on the performance at specific points within the workspace. This distinction occurs because commercial measurements often extend to a global value, reflecting the worst-case performance scenario, whereas standards organizations provide a more detailed local performance assessment [4].

Metric	**Pose dependency**	**Physics**	**Task dependency**	**Sources**
Weight	Global	Dynamics	Intrinsic	[4]
Payload	Global	Dynamics	Intrinsic	[14]
Workspace	Global	Kinematics	Intrinsic	[14, 12]
Speed	Global	Kinematics	Intrinsic	[23]
Position accuracy	Local / Global	Kinematics	Extrinsic/Intrinsic	[14, 24, 4]
Position repeatability	Local / Global	Kinematics	Extrinsic/Intrinsic	[14, 24, 4]
Degrees of freedom	Global	Kinematics	Intrinsic	[25]
Energy/Power consumption	Global	Dynamics	Extrinsic	[22, 26]
Temperature range	Global	Other	Intrinsic	[27]
Joint pose/velocity	Local	Kinematics	Extrinsic	[22, 12]
Actuation latency	Global	Other	Intrinsic	[22]
Pose travel time	Local	Kinematics	Extrinsic	[22]

Table 1.1: Examples of metrics from robot commercial datasheets and standards, adapted from [4].

1.3.2 Kinematic metrics

Our focus is particularly on kinematic metrics because they will be used later to assess the Delta robot's precision, specifically in terms of accuracy and repeatability. As explained in Section 1.2.3, these metrics depend on the robot's geometry, structure, and topology. Although applicable to all types of robots, these metrics

were originally developed for industrial tasks such as manipulation and machining, which require high precision. The following section presents some kinematic metrics in more detail.

Workspace

To begin, as it is usually the first metric to be derived, the workspace is the fundamental metric which is crucial for comparing robot structures. This metric defines the area within the environment that the manipulator's end-effector can reach. When assessing a robot's workspace, intrinsic metrics are frequently employed to calculate workspace volume based on geometrical boundaries such as joint limits, self-collision, etc. Extrinsic workspace metrics are used when the application includes obstacles or external variables that limit the robot's range of motion [4]. In the literature, workspace volumes are classified into different types, with the two main categories being the reachable workspace and the dexterous workspace, although other classifications also exist [12]. Figure 1.1 shows the difference between the reachable and dexterous workspace for a three-joint planar manipulator. Let's give some of its definitions:

- **Reachable Workspace**: It represents the set of reachable points with at least one orientation [18]. It excludes singular points where the robot loses one or more degrees of freedom.
- **Dexterous Workspace**: It is a subset of the reachable workspace where a point can be approached from all directions. More concretely, the end-effector can complete a full 360° rotation about any axis that passes through that point.
- **Workspace Index**: This metric quantifies which locations can be reached by the end-effector of the robot without surpassing any of the robot joints/links limits. The simplest technique to evaluate the workspace is to discretize the objective workspace into nodes[1] and identify which nodes are feasible[2] [17]. In its discrete form it is given as :

$$WSI = \frac{n_{ws}}{n_G} \in [0, 1] \tag{1.1}$$

  where n_G represents the number of feasible nodes and n_{WS} are the nodes in the objective space. The WSI represents the reachable percentage of the objective workspace [17].

[1] Small volumes of the workspace or voxels [4].

[2] Which are the parts of the workspace that the robot is capable of reaching and utilizing.

- **Workspace/Operating volume**: It refers to the amount of space occupied by the manipulator and its links when moving throughout the workspace.
- Other workspaces include the Orientation Angle Workspace, which describes reachable orientations at a given point, and the Partial Orientation Angle Workspace, where a point can be approached by a range of angles less than 360 degrees [12].

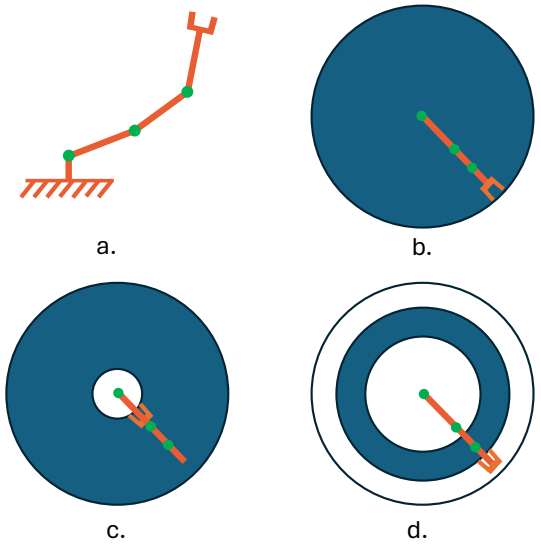


Figure 1.1: Workspace representation, (a) Three-joint planar manipulator, (b) workspace, (c) reachable workspace, and (d) dexterous workspace. Inspired from [28].

Precision metrics

Chapter 4, aims to quantify the precision of a Delta robot. Such robots that performs pick and place or assembling tasks often need high precision capacities [29]. When trying to assess the robot end-effector precision, two indices are often used: repeatability and accuracy. Figure 1.2 illustrates the difference between those two metrics.

- **Repeatability**: "Repeatability is a measure of how close a manipulator can return to a previously taught point. Most present day manipulators are highly repeatable but not very accurate" [30]. This metric is a critical performance metric, particularly for robots performing repetitive tasks like assembly. Typical values of repeatability for industrial robot fall within a range of $\pm 0{,}02[mm]$ to $\pm 0{,}4[mm]$ [31]. For instance, the FANUC LR Mate 200iD/7L has a repeatability of about $\pm 0{,}03[mm]$.

- **Accuracy**: "The accuracy of a manipulator is a measure of how close the manipulator can come to a given point within its workspace" [14] It is the distance [mm] between the reached pose and the desired one.

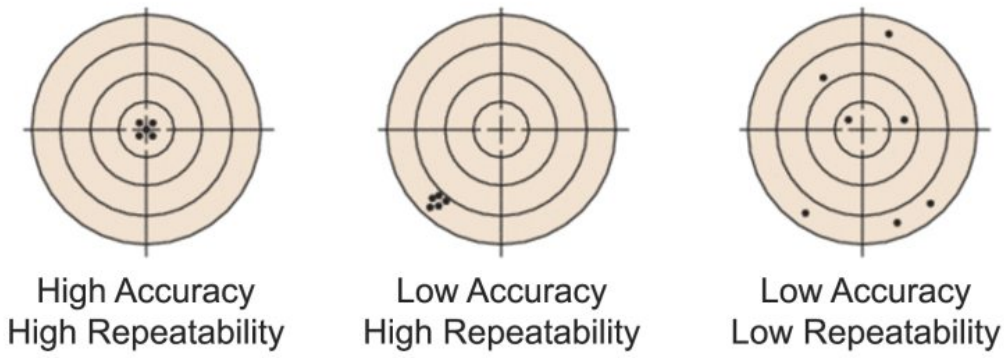


Figure 1.2: "A depiction of the differences between accuracy and repeatability" [32].

Jacobian based metrics

Jacobian-based performance indices are typically local, meaning they depend on the manipulator's posture at a specific location within the workspace. Examples include the manipulability index and the condition number. These kinematic performance indices are derived from the Jacobian, which represents the relationship between joint/actuator velocities and the resulting end-effector velocity [24]. This relationship can be expressed as:

$$\dot{x} = J(q)\dot{q} \tag{1.2}$$

where $J(q)$ is the Jacobian, $q = (q_1, q_2, \ldots, q_n)^T$ is the vector of joint/actuator positions, and $x = (x, y, z, \theta_x, \theta_y, \theta_z)^T$ represents the absolute position and orientation of the end-effector. Moreover, $\dot{q}$ and $\dot{x}$ are their respective time derivatives.

Since many metrics depend on the Jacobian (and on its determinant, singular values, etc.), they are subject to certain limitations. The main ones are listed below :

- **Scale Dependency**: The values derived from the Jacobian are highly dependent on the physical units of measurement used. Different units for link lengths and joint angles can lead to different values, making comparisons across different systems or configurations challenging. It implies that if a metric changes with different units, it is not an absolute measure and can only be interpreted relative to a consistent set of units.

- **Dimensional Non-Homogeneity**: In manipulators that have both prismatic and revolute joints, the Jacobian becomes non-homogeneous due to the different units used for translation $[m]$ and rotation $[rad]$ or $[deg]$ [16]. This makes the evaluation of certain metrics, such as the determinant or singular values, physically inconsistent and difficult to compare or interpret. Therefore, performance indices based on Jacobians are most relevant when the manipulator has the same kinds of degrees of freedom, such as prismatic or revolute, rather than a combination of the two.
- **Frame Dependency**: The Jacobian is not invariant under changes in the reference frame. Consequently, evaluations based on the Jacobian, such as singular values and eigenvalues, can vary with the choice of the reference frame[3].
- **Unbounded Nature of Metrics**: Some performance indices based on the Jacobian matrix, like the manipulability index, are unbounded. This means it can grow without limit, making it a relative rather than an absolute measure of robot performance [12].

Jacobian-based metrics are widely employed for robot mechanism synthesis and performance optimization, and control [4]. As it will be presented, these indices effectively capture robot performance at both local and global levels. However, given these limitations, it is essential to consider them when applying Jacobian-based performance indices to ensure evaluations are both accurate and meaningful. Next, various Jacobian-based indices will be discussed.

- **Manipulability index**: This concept, presented by Yoshikaw [15], is defined as follows for a redundant[4] robot:

$$\mu = \sqrt{\det(J.J^T)} \tag{1.3}$$

It can also be written as follows, using the singular value decomposition :

$$\mu = \sqrt{\lambda_1\lambda_2\lambda_3\ldots\lambda_m} = \sigma_1\sigma_2\ldots\sigma_m \tag{1.4}$$

where λ_i are the eigenvalues of $J.J^T$ matrix and σ_i are singular values of the Jacobian [33].

[3]Except for the Jacobian determinant [12]

[4]A "redundant" manipulator is a robotic system that has more degrees of freedom than are necessary to complete a given task. For an n-DOF robot operating in a m-dimensional task space, the Jacobian J is an $m \times n$ matrix. For a redundant robot, the Jacobian is generally a rectangular matrix with more columns than rows ($m < n$). This implies that the system of equations represented by the Jacobian is underdetermined, meaning there are infinitely many joint velocities $\dot{\theta}$ that can produce the same end-effector velocity $\dot{x}$.

For a non-redundant robot (as the Delta) the Jacobian is a square matrix and the manipulability index is defined as follows [19]:

$$\mu = |\det(J)| \tag{1.5}$$

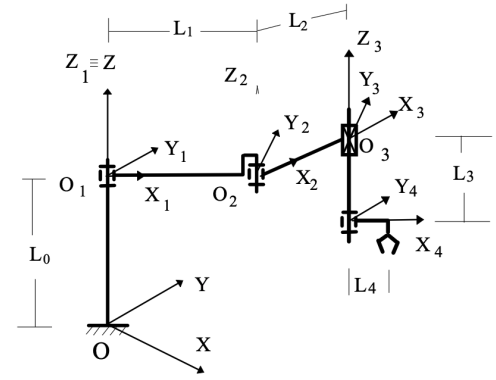
When $\mu = 0$, it indicates a singularity, meaning the system is in a degenerate state where some movements or directions are lost. This is both a necessary and sufficient condition for the presence of a singularity. Consequently, having a higher value of μ is preferable, as it suggests the system is far from singularities and retains full control over all degrees of freedom.

As explained previously in the limitations, the value of the Jacobian matrix determinant depends on the physical units used, therefore it is convenient to use the normalised mobility index written as follows :

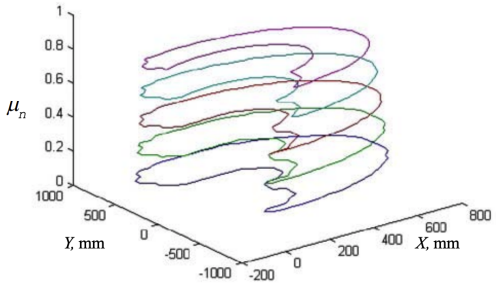
$$\mu_n = \frac{\mu_1 \mu_2 \dots \mu_i}{\max\{\mu_1 \mu_2 \dots \mu_i\}} \tag{1.6}$$

where μ_i is the manipulability index at a given point of the workspace.

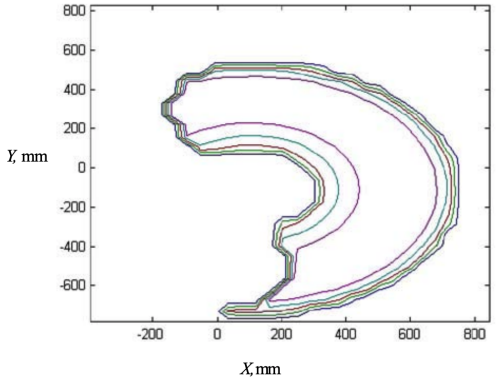
The normalised mobility index is bounded between 0 and 1. Figure 1.3 shows an example of normalised manipulability index over the workspace for constant orientation of a SCARA type robot, from [33]. More concretely, this index measures a robot manipulator's ability to move and apply forces in different directions. In essence, a higher manipulability index indicates better performance and flexibility (maximum value around the center of the workspace), while lower values suggest potential difficulties in achieving desired movements or applying necessary forces, particularly near singular configurations (where it is equal to 0) [34].



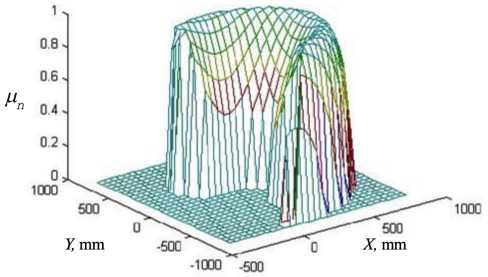
(a) SCARA type robot



(b) 3-D graph of areas with different normalised manipulability index



(c) Areas with different manipulability indices



(d) The normalised manipulability index

Figure 1.3: Example of normalised manipulability index over the workspace for constant orientation of a SCARA type robot, from [33].

- **Condition number**: It is a measure of how well the manipulator can move independently in different directions. More concretely, it assess the degree of independence of the columns of the Jacobian [12].

  Mathematically, the condition number κ of a Jacobian J is defined as the ratio of the largest singular value $\sigma_{\max}$ to the smallest singular value $\sigma_{\min}$ [35]:

  $$\kappa = \frac{\sigma_{\max}}{\sigma_{\min}} \tag{1.7}$$

  However, this formulation can be complex to evaluate as singular values depend on Eigenvalues that are difficult to express analytically [12]. The condition number for an homogeneous Jacobian matrix can be calculated in

a simpler way using the following formula:

$$\kappa = ||J|| \cdot ||J^{-1}|| \tag{1.8}$$

where $||X||$ represents the norm of the matrix X. The definition of the condition number dependent of which matrix norm is considered. The main used are:

- The Frobenius (also called Euclidean) norm, which allows to express κ as:

$$\kappa = \frac{1}{n}\sqrt{\text{tr}(JNJ^T)\text{tr}(J^{-1}NJ^{-T})} \tag{1.9}$$

  where tr is the trace and matrix N is given as:

$$N = \frac{1}{n}I_{n\times n} \tag{1.10}$$

  where I is the identity matrix and n is the dimension of the square matrix.[12].

- The 2-norm (also known as the spectral norm), which for a matrix A is its largest singular value (the square root of the largest eigenvalue of matrix $A^{-T}A^{-1}$) [36].

Numerical analysts have shown that the condition number is a more accurate indicator of the robot's ill-conditioning[5] than the manipulability index [12]. However, even though it addresses some limitations of the manipulability index (e.g. can assess robot with both prismatic and revolute joints), the condition number still has drawbacks due to its reliance on the determinant of the Jacobian.

Its value ranges from 1 to ∞. Therefore, it is more convenient to use its reciprocal, that would represent an "ideal" relative index [4], which is bounded between 0 (for singular configurations) and 1 (for the best performance). Figure 1.4 illustrates the reciprocal condition number of a SCARA type robot from [33].

[5]Ill-conditioning refers to situations where small changes in the input or joint positions of the robot lead to disproportionately large changes in the output or end-effector position and orientation. An ill-conditioned robot is difficult to control accurately and might behave unpredictably, especially when doing tasks that need a lot of precision.

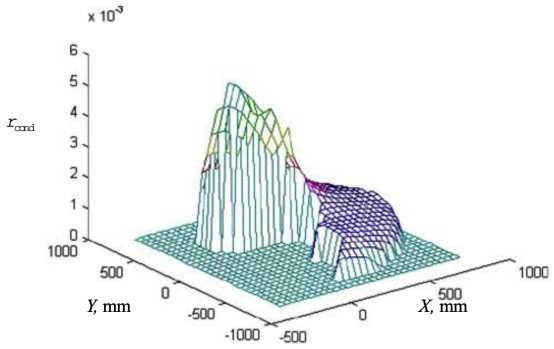


Figure 1.4: The reciprocal condition number of a SCARA type robot [33].

Those are local metrics. However, as explained previously 1.2.2, local metrics can be converted to global scale by integrating local metrics across the configuration space. Therefore, by integrating the reciprocal condition number over the entire workspace W, one gets a new global metrics : **Global conditioning index**. The latter one can be expressed as [4, 12]:

$$\eta = \frac{\int_W \kappa^{-1} dW}{\int_W dW} \tag{1.11}$$

which corresponds to the mean value of $1/\kappa$ over the entire workspace [36]. Its value ranges from 0 to 1. However, due to the challenges associated with solving the integral term exactly, a discrete approach is often employed instead[17].

One can similarly use other local indicators to obtain the average index within the target workspace volume, thereby characterizing the overall performance of the robot.

1.3.3 Static metrics

These indices measure how a robot performs when carrying a load, and they are usually applied in slow tasks with limited acceleration.

- **Stiffness matrix**: The flexibility of the Delta robot's distal arms will be experimentally evaluated in this study. This assessment can also be conducted through simulations, such as Matrix Structural Analysis (MSA), which was utilized in the previous thesis [3]. An overview of the MSA model can be found in [37], which employs a static metric known as the Stiffness Matrix.

  The stiffness matrix is a crucial concept in robotics that helps understand and measure how a robot's structure responds to applied forces and torques. It

quantifies the robot's resistance to deformation, ensuring precise control and operation [14]. There are two main types of stiffness matrices: the Cartesian stiffness matrix (K_C) and the joint-space stiffness matrix (K_θ).

The Cartesian stiffness matrix evaluates the deflection of the robot's end-effector when a wrench[6] is applied. It is represented by the equation:

$$f = K_C \Delta x \tag{1.12}$$

where f is the applied wrench, K_C is the Cartesian stiffness matrix, and Δx is the resulting displacement of the end-effector.

On the other hand, the joint-space stiffness matrix computes the deflection of the robot's joints when a torque (rotational force) is applied on the end-effector. It is represented by the equation:

$$\tau = K_\theta \Delta q \tag{1.13}$$

where τ is the applied torque, K_θ is the joint-space stiffness matrix, and Δq is the resulting deflection of the joints.

These two matrices are interconnected through the Jacobian. The combined relationship is given by [38]:

$$K_C = J^{-T} K_\theta J^{-1} \tag{1.14}$$

The stiffness matrix is frequently used to characterize the elastic response of a robot's end-effector to applied loads [4]. Be aware that this formulation also suffer from the limitations associated with the Jacobian, previously discussed here 1.3.2.

The stiffness matrix is essential for performance assessment, helping evaluate the robot's ability to resist deformation, which is critical for tasks requiring

[6]A wrench is a 6-dimensional vector that combines both the force and the moment (or torque) acting on a point, typically at the end-effector of a robot. It can be represented as:

$$\mathbf{w} = \begin{bmatrix} \mathbf{f} \\ \mathbf{m} \end{bmatrix}$$

where:

- $\mathbf{f}$ is the force vector, consisting of the components f_x, f_y, and f_z, which represent the forces acting along the x, y, and z axes, respectively.
- $\mathbf{m}$ is the moment (or torque) vector, consisting of the components m_x, m_y, and m_z, which represent the torques about the x, y, and z axes, respectively.

high precision and accuracy. It also aids in design optimization, allowing engineers to design robots with the appropriate stiffness to handle specific loads and forces without significant deformation.

- **Payload indices**: This metric measures how well a robot manipulator can transmit force. It helps determine the maximum or minimum loads that the robot's end-effector can handle.

  The relationship between the input forces or torques in the joints (τ) and the output forces or load at the end-effector (F) is given by

  $$\tau = J^T F \tag{1.15}$$

  where J^T is the transpose of the Jacobian.

  This metric evaluates the manipulator by using the maximum singular value (λ_P) of J^T to quantify the maximum payload capacity ($\|F_{\text{Max}}\|$) for a given configuration of the robot, calculated as

  $$\|F_{\text{Max}}\| = \sqrt{\max(\lambda_P)} \tag{1.16}$$

  This capacity is pose-dependent since the Jacobian changes with the robot's posture.

  To determine the maximum payload capacity within the manipulator's entire workspace, the global payload index integrates the local payload indices over the workspace, expressed as

  $$P_{\text{Max}} = \frac{\int_W \|F_{\text{Max}}\| \, dW}{\int_W dW} \tag{1.17}$$

  in [14]. This equation provides an average payload capacity over different points within the workspace, offering a comprehensive measure of the manipulator's capability. In this context, the payload index is considered a static metric. However, if it were redefined to assess the maximum load a robot can handle while in motion, taking into account acceleration and inertial forces,it would then be classified as a dynamic metric.

1.3.4 Dynamic performance metrics

- **Dynamic manipulability index**: It measures the ability of the robot to generate acceleration based on a joint driving force [12]. It consists basically of the kinematic manipulability index (previously introduced in 1.3.2) to

which was added the inertia matrix M[7] of the system. Yoshikawa [15] defined it as:

$$\mu_d = \sqrt{\det[J(MM^T)^{-1}J^T]} \tag{1.18}$$

As its kinematic version, it is also a pose-dependent local performance metric, and therefore does still suffer from the same limitations.

- Other dynamic indices also exist such as the Dynamic Dexterity Index and the Dynamic Conditioning Index.

1.4 Conclusion

This state-of-the-art introduced the notion of *performance metrics*, reviewed different classifications used to categorize those metrics in the literature, and presented some performance indices in more detail. These metrics are essential tools for comparing different robotic systems, optimizing their designs, and improving their operational efficiency through control strategies.

In the previous thesis on the Delta robot [3], various performance indices were utilized to assess and enhance the robot's design. These indices played a crucial role in evaluating key aspects of the robot's functionality, helping to ensure efficient operation and optimal performance.

In this master thesis, the focus is not on the design but on the evaluation of some of the Delta robot performances, including pose accuracy and repeatability. This assessment is presented in Chapter 4.

[7] M represents the inertia matrix that is used in the dynamic equation of a robotic manipulator (adapted from [39]):

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + N(q) = \tau$$

where $C(q,\dot{q})$ represents Coriolis and centrifugal effects, $N(q)$ the potential forces, and τ is the joint torque. The inertia matrix M thus plays a crucial role in determining how effectively the robot can accelerate in response to applied torques.

Chapter 2

Robot Design Amelioration and Implementation

The main functions and constraints for the Delta robot were identified in the previous thesis [3]. The goal remains to demonstrate the robot's advantages and parallel structure through various movements and actions, while ensuring safety for a young audience such as that of the "Printemps des Sciences" event of Louvain-la-Neuve [40]. It is essential to revisit and review these original constraints (Appendix A), identify any new ones, explain their necessity, and assess adherence to previous constraints. Once established, the improvements made to the robot will be detailed.

2.1 Prior Functions and Constraints Review

MF1: Accuracy and Repeatability

The robot's required accuracy was set at $1[mm]$, with repeatability crucial for tasks like pick and place specified to be within $\pm 0.5[mm]$. Precision tests, detailed in Chapter 4, confirmed these constraints were met, achieving even better precision and repeatability.

MF2 & MF3: Payload, Speed, and Acceleration

The requirements for maximum payload ($1[kg]$ in static and $250[g]$ in dynamic), speed ($3[m/s]$), and acceleration ($10[g]$) were satisfied last year, so no changes have been made.

MF4: Task Performance

The Delta robot must perform a variety of tasks to fulfill its educational role. While not fully accomplished in the previous thesis, the robot successfully executed:

1. Pick and Place trajectory
2. Circle trajectory
3. Helix trajectory

We aim to expand on these trajectories and explore additional tasks, which will be detailed in the following section.

MF5: Replaceability

The different parts, mechanics and electrical, of the robot must be easily replaceable in case of failure. This function was already satisfied and required no further action.

MC: Last Year Constraints

The constraints outlined in [3], can be found in Appendix A. They remain unchanged and require no additions.

2.2 Additional Main Functions

MF6: Accuracy and Flexibility Validation

The Delta robot, due to the stiffness of its parallel kinematic chain, exhibits high precision. Last year, a numerical validation using Robotran[1] [41] supported this claim. This year's MF1 aims to confirm the numerical model and quantify the robot's precision and flexibility.

MF7: Graphical User Interface

The Delta Robot should be controlled through an easy-to-use interface.

MF8: End Effector Capabilities

The end effector must grab and release items, carrying at least $1[kg]$ statically and $250[g]$ dynamically.

[1]Robotran is a powerful multibody dynamics simulation software that allows for the modeling and analysis of mechanical systems, providing detailed insights into their dynamic behavior.

MF9: Task Performance / Demonstration

As previously mentioned, the Delta robot must perform a variety of tasks and demonstrations to fulfill educational purpose. Here are some examples:

- Pick and Place tasks
- High speed and precision trajectory
- Highlighting the critical role of motion laws

2.3 Additional Main Constraints

No new constraints were added this year; however, those mentioned in [3] still need to be respected. The most notable one being usability: The robot must be reliable for long-term use, with no risk of overheating or mechanical failure for at least 10 years. It should also be easy to operate, with clear documentation provided. Additionally, the design must allow an unobstructed view for filming and viewing during sessions.

With the objectives clarified, the focus shifts to explaining how MF6 was addressed. To enhance functionality and make the pedagogical demonstration more tangible, a pneumatic gripper with a suction cup system was integrated into the end effector, along with an electro-magnet gripper. The following sections provide details on both grippers.

2.4 Pneumatic dimensioning

In order to select the appropriate components for this first gripper, a pneumatic dimensioning as been made. The latter one follows the following structure :

1. Defining the forces and constraints of the application,
2. Determining the suitable suction cups,
3. Determining the suitable vacuum generator,
4. Selecting accessories for the control.

The detailed budget of the chosen components is available in Appendix B.

2.4.1 Defining the forces and constraints of the application

According to the specifications, as stated in Section 2.1, the robot should at least be able to carry a $1[kg]$ mass in static and a $250[g]$ under dynamic conditions.

In order to compute the necessary holding force to carry a moving payload, two different situations need to be considered [42]. The first case being with an horizontal vacuum suction cup and a vertical direction of force, as shown in Figure 2.1. And the other with an horizontal direction of force and a horizontal vacuum suction cup, as illustrated in Figure 2.2. These are the only scenarios that need to be considered because the end-effector of the Delta robot always remains in a horizontal orientation.

In the first case, we can calculate the theoretical holding force (F_H) as follows:

$$F_H = m \times (g + a) \times S \tag{2.1}$$

where:

- F_H = Theoretical holding force at the suction cup (N),
- m = Mass of the object (kg),
- g = Acceleration due to gravity (m/s^2),
- a = Acceleration of the object (m/s^2),
- S = The safety factor (at least 2.5 for high acceleration tasks [42]).

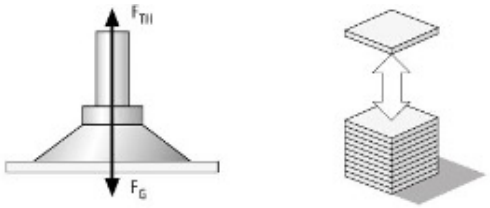


Figure 2.1: Suction cup horizontal, direction of force vertical, from [43].

In the second scenario, where the force is applied horizontally, as illustrated in Figure 2.2, it is necessary to account for the friction force. This is done by considering the relationship between the friction force and the normal force, which leads to the following expression:

$$F_H = m \times (g + \frac{a}{\mu}) \times S \tag{2.2}$$

where, μ is the friction coefficient.

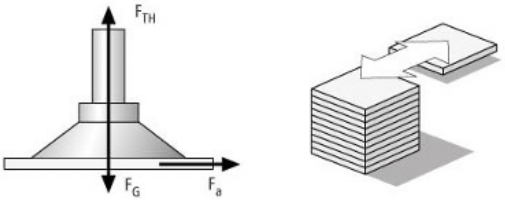


Figure 2.2: Suction cup horizontal, direction of force horizontal, from [43].

Therefore, assuming the specifications require a $1[kg]$ mass in static, the following holding-force would be needed:

$$F_H = m \times (g + a) \times S = 1 \times (9,81 + 0) \times 1,5 = 14,715[N] \tag{2.3}$$

For the dynamic scenario, the second case is selected because it requires a greater holding force than the first. The Delta robot is intended to serve as a demonstrator, capable of performing manipulations of varying amplitudes. However, it has a limited workspace, as detailed in Chapter 4. To accommodate this, 3D-printed cylindrical objects were designed, as shown in Figure 2.3. These cylinders, measuring $5[cm]$ in diameter and $2[cm]$ in height, were chosen to enable various manipulations within the workspace. The parts are printed in PLA, which has a friction coefficient of 0.5 based on [44], making them significantly lighter than $250[g]$ (the load the Delta should be able to carry under dynamic conditions), with an approximate weight of $17[g]$.

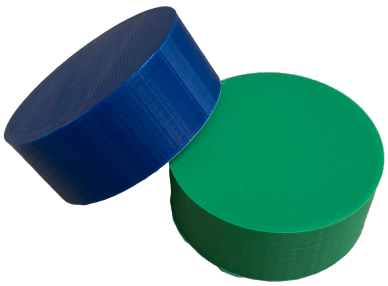


Figure 2.3: 3D-Printed cylindrical objects for the demonstration.

The robot is built to handle this payload ($250[g]$), but because it will rarely need to do so in practice, arbitrary acceleration value of $3[g]$ is used for calculations. This value, while below the robot's maximum capacity, provides a holding force sufficient for our requirements. By designing the system around a $3[g]$ acceleration with a $250[g]$ payload, the robot can operate at full acceleration when using the

specific objects of the demonstrations. This ensures that the holding force is more than adequate for the tasks, while also effectively respecting the specifications. With these considerations, one gets the following force:

$$F_H = m \times (g + \frac{a}{\mu}) \times S = 0,25 \times (9,81 + \frac{(3 \times 9,81)}{0,5}) \times 2,5 = 42,92[N] \quad (2.4)$$

which is way superior than the static holding-force (almost 3 times superior).

2.4.2 Determining the suitable suction cups

When selecting the appropriate suction cups, it's crucial to first assess the specific application requirements, including the number of suction cups needed. The number of cups varies depending on the object being handled and the surface area involved. The idea is to select a number of suction cups that would prevent the surface of the object used for the suction to bend (avoiding air leakage). Finally the diameter of the suction will change according to the number of cups as the load distribution varies according to it [43].

In the Delta robot demonstrations, which uses specific 3D-printed cylindrical objects, as previously explained, a single suction cup is sufficient to securely handle each object.

The necessary suction cup diameter is directly related to the holding force required. Although methods such as using a selection graph (Figure 2.4) or calculations based on the formula from [45] are available, in this case, the cup size was chosen based on existing specifications provided by Festo:

$$d = 1.12 \times \sqrt{\frac{m \times S \times \left(g + \frac{a}{\mu}\right)}{(P0 \times n)}}$$

where P_0 is Vacuum in [bar] and n the number of suction cups.

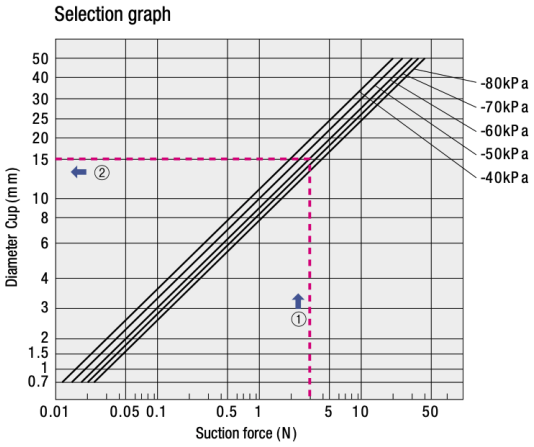


Figure 2.4: Selection graph from Misumi.

Various types of suction cups are designed to handle specific surfaces and materials, as illustrated in Figure 2.5. The lip suction cup was selected due to its sealing lip, which prevents leaks and ensures a reliable seal on smooth or slightly rough surfaces. This feature is especially effective at maintaining vacuum, making it ideal for precise and secure handling of objects in this application.

(a) Lip suction cup (b) Oval suction cup (c) Bellows suction cup (d) Flat suction cup

Figure 2.5: Different types of suction cups, from [45].

With these parameters carefully determined, including the type of suction cup, and required holding force, the selection process leads to the selection of the following suction cup from Festo, shown in Figure 2.6. With a 30[mm] diameter, the latter one generate an holding force of 36[N] when working under nominal operating pressure (which is -0,6[bar]). It also includes a 3[mm] height compensator, which allows for slight variances in the height of the object it is working with. This suction cup is ideal for the cylinders, as it is designed to handle objects with a minimum radius of 35[mm], while one of the cylinders has a radius of 50[mm].

(a) (b)

Figure 2.6: (a) OGVM-30-S-N-G14F Lip suction cup from Festo and (b) its technical drawing.

2.4.3 Determining the suitable vacuum generator

To generate a vacuum, it is not always necessary to use a vacuum pump. Indeed compressed air can economically produce a vacuum through ejectors that operate based on the Venturi principle, shown in Figure 2.7. Since vacuum suction is often required only for a short period in automated machines, using this type of vacuum generator can be more advantageous than using a vacuum pump. Additionally, the control of these systems is very flexible and fast, which is often a requirement in automation.

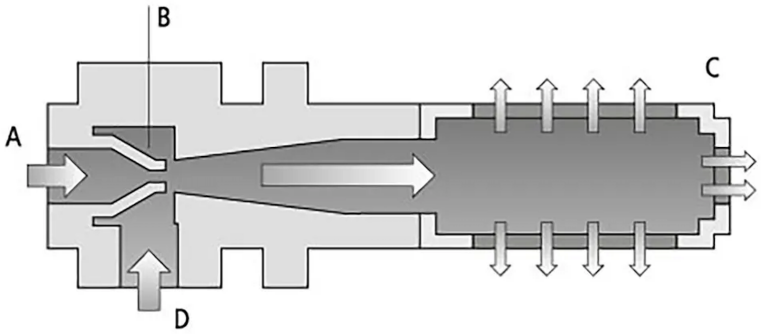
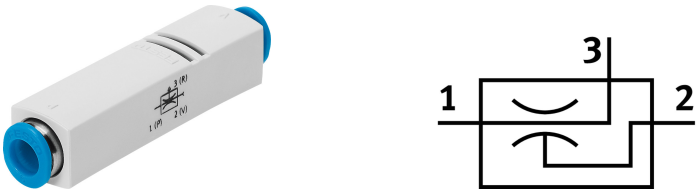


Figure 2.7: Illustration of the Venturi principle as applied in pneumatic vacuum generators. Compressed air is introduced into the ejector (A), where it is accelerated through the Venturi nozzle (B), causing a reduction in static pressure and the generation of a vacuum. The air is then drawn through the vacuum connection (D) and expelled along with the compressed air through the silencer (C)[43].

Here are some advantages of using pneumatic vacuum generators:

- High vacuum capacity with low air flow rates,

- No wear and tear or maintenance required for the components,
- Cost-effective,
- Compact design and lightweight,
- Mounting position can easily be adapted to the user's preference.

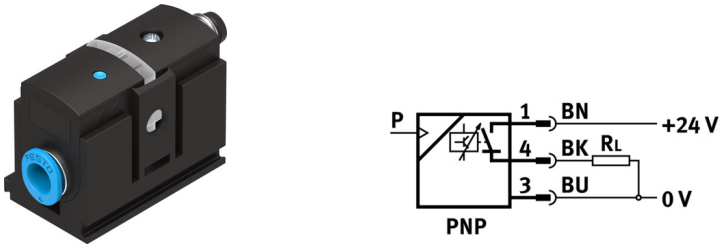


(a) (b)

Figure 2.8: (a) VN-05-M-I3-PQ2-VQ2 Vacuum generator from Festo. and (b) its associated technical drawing with: 1. Where the compressed air enters, 2. Where the vacuum is generated, and 3. The exhaust port where the air exits after generating the vacuum.

2.4.4 Selecting accessories for the control

Between the vacuum generator and the gripper, additional components can be placed depending on the application. For this task, in order to ensure that the objects are securely gripped by the suction cup, it is necessary to measure and monitor the vacuum at the suction cup. To do this a pressure sensor can be used to instantly provide information about the actual pressure (allowing confirmation of suction in vacuum technology, or object detection via back pressure). The chosen pressure sensor is shown in Figure 2.9.



(a) (b)

Figure 2.9: (a) SDE5-V1-FP-Q6-P-M8 Pressure sensor and (b) its associated technical drawing, from Festo.

It has been configured in fixed hysteresis threshold comparator mode, whose

diagram is shown in Figure 2.10. In this mode, the sensor acts as a comparator using two learned (preset) pressure values. With these settings, the sensor's LED lights up when the pressure in the cup is sufficient to securely hold the object it contacts.

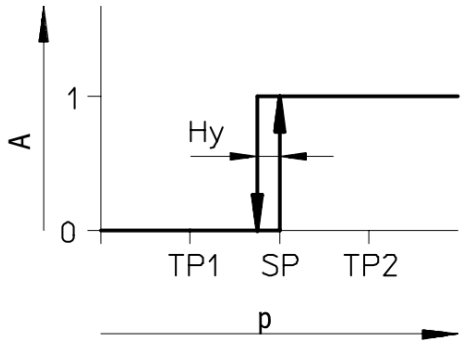
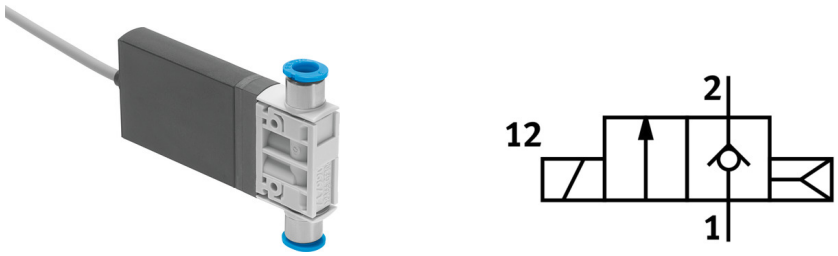


Figure 2.10: Diagram of the operating mode of the pressure sensor from Festo. TP1: the pressure when the suction cup is not in full contact with an object, and TP2: when the pressure in the cup is good enough for the object to be handled.

Additionally, a valve is needed to control the flow of compressed air for operating the suction cup and handling the objects. The chosen valve, shown in Figure 2.11a, operates at 24$[V]$, requiring a power supply capable of delivering at least 1.7$[A]$ without voltage drop. Since the built-in I/O points of the ClearCore cannot provide sufficient current, the valve's power cables (V+ and GND) were connected directly to the ClearCore's power supply, which can supply 24V and 6.5A. However, the control cable was connected to an I/O point on the ClearCore, as illustrated in the electrical diagram in Figure **??**.



(a) (b)

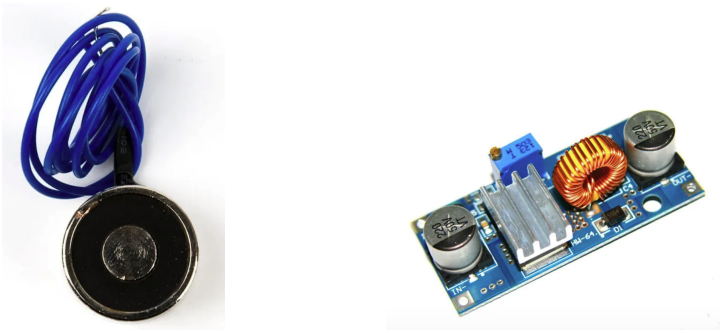
Figure 2.11: (a) MHJ10-S-2,5-QS-6-HF/LP Solenoid valve and (b) its associated technical drawing, from Festo.

During the implementation and calibration of the pneumatic system, an issue was

identified with the valve: it occasionally fails to close when mounted vertically. Although it is only an hypothesis, this may be due to insufficient power to counteract the combined effects of pressure and gravity on the spool. Interestingly, if the pressure is briefly released and then reapplied, the valve closes correctly. When the valve is positioned horizontally, it functions properly without these issues.

2.5 Electro-magnet

To facilitate and accelerate the development of the code to control the suction cup before the arrival of the pneumatic materials, an electromagnet was employed as a substitute. This choice was made because both systems require similar control logic, simply turning an electrodistributor or the electromagnet on or off. The electromagnet, though compact, generates a theoretical maximum force of 30$[N]$, with actual force varying based on surface conditions and metal type. Operating at 12$[V]$ and 0.2$[A]$, it engages when powered and releases when de-energized, making it a good option for initial testing.



(a) (b)

Figure 2.12: (a) Electromagnet DC12V 3KG (OT5246) and (b) Adjustable Step-Down Module XL4015 (OT2004-A113), both from Otronic.

Since the electromagnet operates at 12$[V]$ while the ClearCore is powered at 24$[V]$, a buck converter was used to step down the voltage to the appropriate level, the corresponding connections are shown in Figure **??**.

Chapter 3

Robot Interface

3.1 Introduction

To improve the usability of the Delta robot, a Graphical User Interface (GUI) was developed. Previously, controlling the robot's trajectories required manual changes to the C++ code within Atmel/Microchip Studio, followed by flashing the new firmware to the robot's micro controller (ClearCore). This process was complex and not user-friendly, especially for those without programming skills.

The new GUI was designed to simplify this process by allowing users to control the robot and launch trajectories without needing to modify the code. This interface provides a visual, intuitive way to use the Delta robot, making it accessible to users with varying technical backgrounds.

3.2 Design Considerations

The interface was designed with a focus on usability and robustness. Upon launching, users should easily understand how to use the Delta robot properly (being able to launch sequences). To ensure this, a message with detailed instructions appears when the interface is opened, as shown in Figure 3.1.

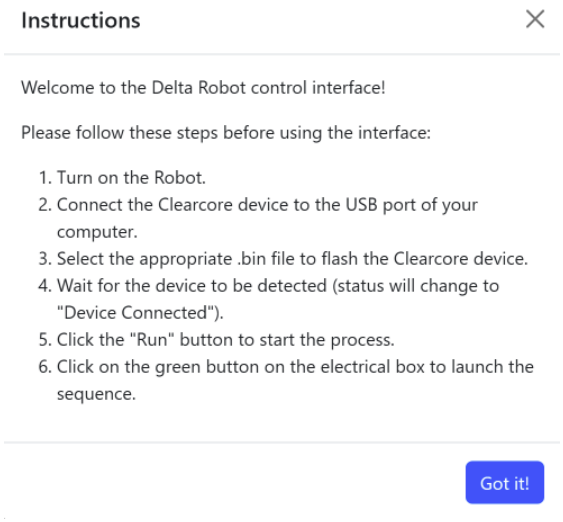


Figure 3.1: Opening instruction message of the GUI.

Because some communication is required between the computer and the robot, one have to wait for the USB Serial Port to be found (meaning both devices are well connected) before launching any sequence. In order for the user to know when this is possible, a *device connection status* displays whether the communication is possible or not. For example, when the device is not connected, the message reads **Status: Device Not Connected**, as shown in Figure 3.2. When no device is connected it's impossible for the user to click on any button that would allow to start a sequence.

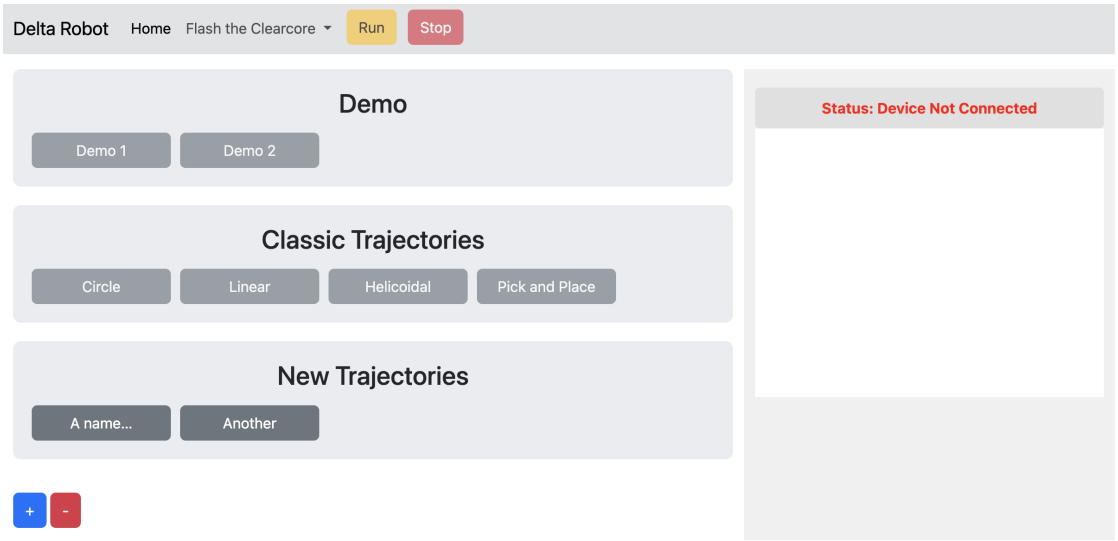


Figure 3.2: Graphical user interface to control the Delta robot.

While the GUI does not allow users to create new sequences (trajectories), it does provide a feature that lets users add a new button by clicking the + icon. This new button can be configured to launch a different trajectory by assigning it a new .bin file[1]. If one only wants to flash some code without creating new button, on can use the dropdown button in the menu to flash a .bin file directly. Finally, pressing the run button allows to start a sequence that already resides in the memory of the ClearCore. When a file is flashed the progression of the flashing process is shown in the terminal window on the right on the interface, see Figure 3.2.

3.3 Architecture and Implementation

The interface has been implemented using ElectronJS [46]. This framework enables the development of cross-platform desktop applications that run on Windows, macOS, and Linux using web technologies such as HTML, CSS, and JavaScript. It is used by thousands of companies to build cross-platform software [2]. The front-end is rendered using *Chromium*, while the back-end is powered by *Node.js* [47].

- Chromium is an open-source web browser project developed and maintained by Google. It serves as the foundation for several web browsers, most notably Google Chrome.

[1]To get the .bin file associated with a new trajectory, the user still needs to use Atmel/Microship studio as explained in appendix ...

[2]Among which : Discord, Dropbox, Notion, Figma, Skype, Slack, Twitch, Microsoft Teams, VS Code, and way more...

- Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to execute JavaScript code outside of a web browser.

3.3.1 Multi-process model architecture

A classical Electron project is structured around a few key files (including a *main* script and several *renderer* processes). The *main.js* controls the application's logic and can instantiate numerous renderer processes [47]. It is responsible for managing the application's lifecycle, such as opening windows, handling system events, and managing global data. Each renderer process is isolated, running in its own instance of Chromium, which enhances security and stability by preventing crashes in one process from affecting others.

Despite this isolation, communication between the main process and renderer processes is essential for the application's functionality, which is managed through Inter-process Communication (IPC), discussed in the following section. Therefore, one can say that it works as a multi-process architecture[3].

3.3.2 Inter-process communication (IPC)

Inter-process communication (IPC) in Electron enables the main process and renderer processes to communicate effectively. It allows to coordination of tasks between the processes, such as executing commands in the main process based on user actions in the renderer.

The IPC can either be *synchronous* or *asynchronous*.

- Synchronous IPC: This method allows the renderer process to send a message to the main process and wait for an immediate response. While useful for scenarios requiring instant feedback, synchronous IPC can block the renderer process, potentially affecting performance.
- Asynchronous IPC: More commonly used, asynchronous IPC allows the renderer process to continue running while awaiting a response from the main process. This approach is more efficient and prevents UI freezes, making it the preferred method for most communications.

In the interface, IPC is implemented using the ipcMain module in the main process, which listens for events, and the ipcRenderer module in the renderer process, which sends messages. For example, if a user clicks on a button to flash some code to the ClearCore, the renderer process will request the main process to execute the

[3]This kind of architecture has been developped for webbrowsers to prevent one website to affect the entire browser if it happened to crash.

action, and the main process will then send back a confirmation or error message. These methods allow the renderer process to continue running while waiting for a response from the main process, preventing any UI freezes and ensuring a smooth user experience.

3.3.3 Flashing process

The most important aspect to be able to "get rid" of microship studio, is to implement a flashing process. As the ClearCore works with an ARM microcontroller (32-bit ARM Cortex M4F processor), one can use *BOSSA*, a flash programming utility for Atmel's SAM family [48].

To make this work a batch file[4] (.cmd) format which automates the flashing process using the BOSSA utility has been written. Firstly, it validates type of input file (binary), then locates the required BOSSA tool. For the next step, it tries to identify the correct COM port for the ClearCore device until it succeed, and then uploads the firmware.

The script ensures that the ClearCore is in bootloader mode before flashing. "A bootloader is a small piece of code that allows users to reprogram the application code within that device" [49]. In this mode, the device is ready to receive new firmware, and the existing application is not currently running. Finally, this script handles any errors that may occur during the process.

3.3.4 Control

The control of the Delta robot is divided into two main parts. The first part is the non-real-time phase, where controlling the robot requires modifying or writing C++ code. This code is responsible for defining the robot's trajectory and for monitoring and updating the states of external devices (such as microswitches, the Venturi system, and buttons). Once the code is written and compiled, it is sent to the ClearCore, marking the beginning of the second part: the real-time phase.

In the real-time phase, every action directly influences the robot's state. The ClearCore manages the motors' angular positions and continuously monitors and updates the status of external devices through the I/O module. This setup ensures secure control via microswitches and enables responsive control for tasks like pick and place through the Venturi system. Although the robot's trajectory is predefined and cannot be adjusted in real time (since it is coded in the non-real-time phase),

[4]Batch file: text file containing a series of commands that are executed by the Windows Command Prompt. So because it uses special Windows prompts it only works on windows for the moment.

the robot can still stop its motion if a microswitch is triggered. Additionally, actions can be executed via button presses, allowing real-time interaction with the robot, made possible by the I/O module.

A schematic of this control structure is shown in Figure 3.3.

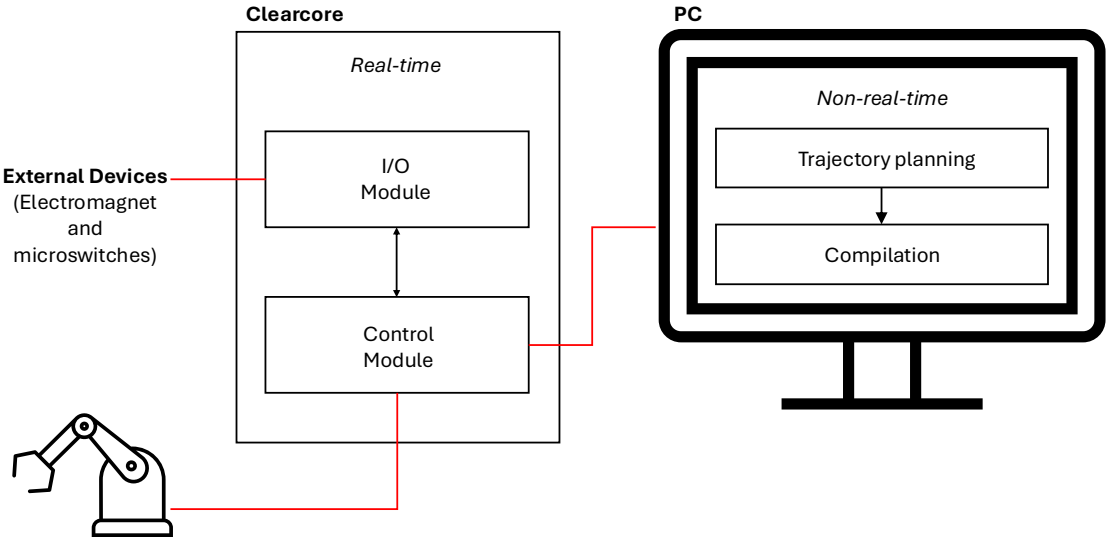


Figure 3.3: Delta robot's controller schematic

The interface eliminates the need for the non-real-time part of the controller, saving significant time by directly flashing pre-compiled files onto the ClearCore. However, while the interface simplifies many aspects, adding a new sequence still necessitates writing or modifying C++ code. This requirement ensures flexibility and customization but requires some programming effort.

The current code begins by initializing the ClearCore board, defining all ports within the code to enable subsequent communication. Next, it sets up the different trajectories variables (e.g., the radius of a circular trajectory) and configures the motors, notably their speed and torque limits. Following this, the code generates an array of positions for the three motors by selecting the desired trajectory, computing each point at time t, and calculating the inverse kinematics. The computed positions are then sent to the motors, while the states of various external devices are continuously monitored.

The structure of this C++ code is illustrated in Figure 3.4, where it is summarized in a diagram for clarity.

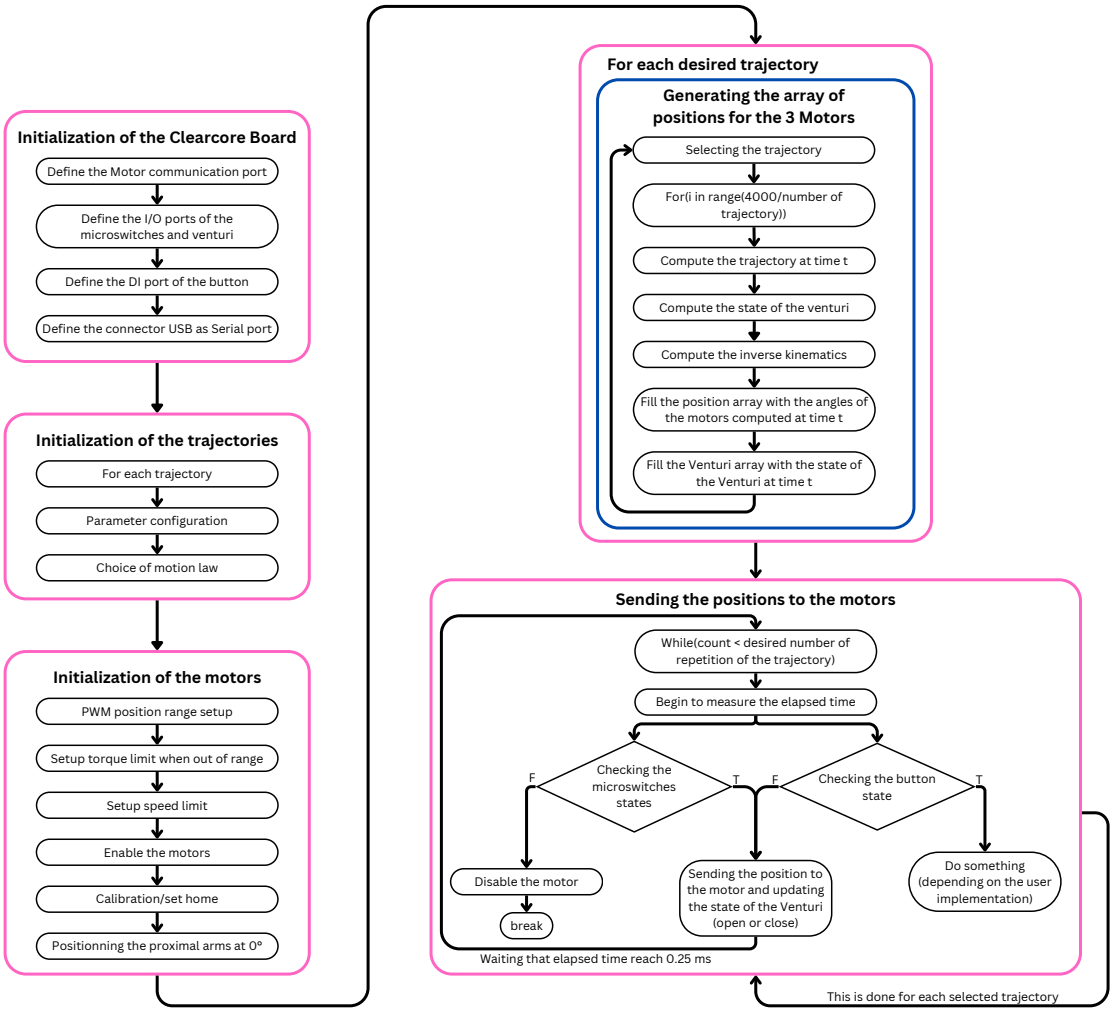


Figure 3.4: Detailed diagram of the robot's code structure

Chapter 4

Accuracy and repeatability testing

This chapter presents the tests that have been carried out on the robot in order to quantify its precision, building on the knowledge from Chapter 1. As outlined in Chapter 2, the required accuracy is $1[mm]$, with repeatability within $\pm 0.5[mm]$. The goal of this chapter is to assess whether or not the robot meets these specifications.

4.1 ISO norm

When attempting to evaluate a technology that has been well-developed in the industry, it is good and common practice to use an ISO norm. Since the purpose is to evaluate the precision of an industrial robot, the most adequate norm is the 9283 one entitled: Manipulating industrial robots — Performance criteria and related test methods [9].

4.1.1 Test Poses and Paths

Following the norm, one must start by locating a single cube, the corners of which are designated C_1 to C_8 (see figure 4.1), in the working space. The cube needs to assure two criterion as stated in [9]:

- the cube shall be located in that portion of the working space with the greatest anticipated use
- the cube shall have the maximum volume allowable with the edges parallel to the base coordinate system

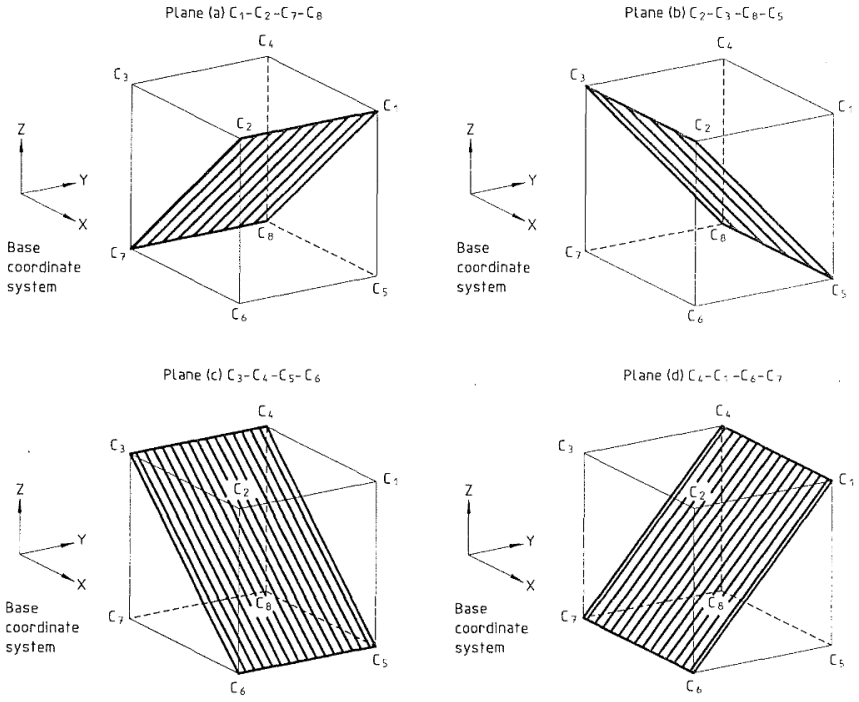


Figure 4.1: Cube within the working space [9].

Once the cube is well defined, one of the following planes shall be used for pose testing:

1. C_1 - C_2 - C_7 - C_8
2. C_2 - C_3 - C_8 - C_5
3. C_3 - C_4 - C_5 - C_6
4. C_4 - C_1 - C_6 - C_7

Still following the norm, five measurement points (P_1 to P_5) are located on the diagonals of the measuring plane. P_1 is the intersection of the diagonals and is the centre of the cube. The points P_2 to P_5 are located at a distance from the ends of the diagonals equal to (10 ± 2) % of the length of the diagonal (see Figure 4.2).

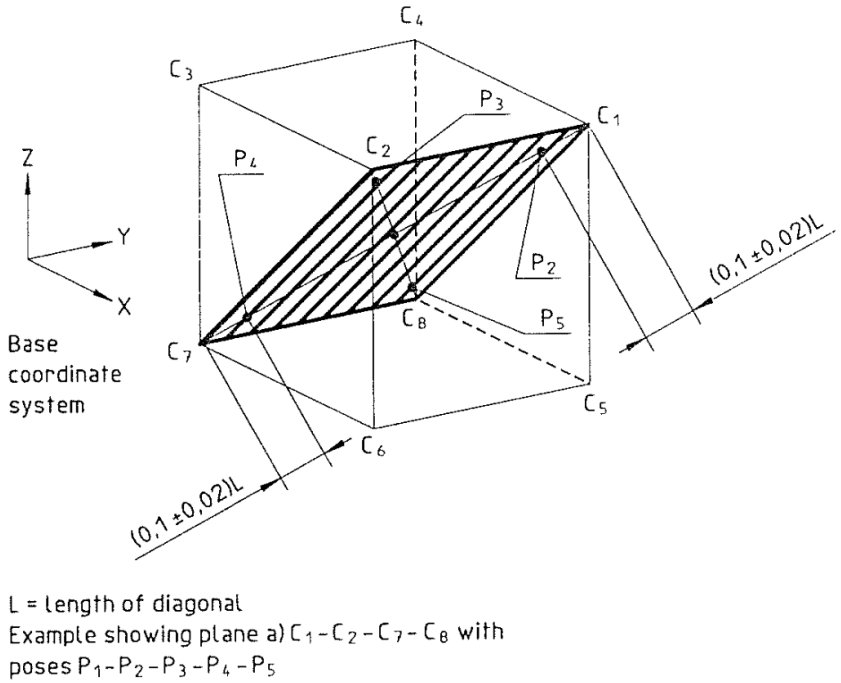


Figure 4.2: Poses to be used for precision measurement within the cube [9].

Once those points are well defined, one can start testing for the pose accuracy and repeatability of his robot.

4.1.2 Pose accuracy

Pose accuracy expresses the deviation between a command pose and the mean of the attained poses when approaching the command pose from the same direction [9]. Its mathematical definition is expressed as follows:

$$AP_P = \sqrt{(\bar{x} - x_c)^2 + (\bar{y} - y_c)^2 + (\bar{z} - z_c)^2} \tag{4.1}$$

with

$$\bar{x} = \frac{1}{n}\sum_{j=1}^{n} x_j$$

$$\bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_j$$

$$\bar{z} = \frac{1}{n}\sum_{j=1}^{n} z_j$$

$\bar{x}$, $\bar{y}$, and $\bar{z}$ are the coordinates of the barycentre of the cluster of points obtained after repeating the same pose n times.
x_c, y_c and z_c are the coordinates of the command pose.
x_j, y_j and z_j are the coordinates of the j-th attained pose.

Load	Velocity	Poses	Number of cycles
100% of rated load	100% of rated velocity 50% of rated velocity 10% of rated velocity	$P_1 - P_2 - P_3 - P_4 - P_5$	30

Table 4.1: Summary of test conditions for pose accuracy [9].

Table 4.1 provides a summary of test conditions for pose accuracy (and also for pose repeatability, which will be discussed in the next section). Starting from P_1, the robot successively moves its mechanical interface to the poses P_5, P_4, P_3, P_2, and P_1. Each of the poses should be visited using a unidirectional approach, as shown by either of the cycles illustrated in Figure 4.3. The second approach, which was easier to implement using only three lasers, was chosen. For each pose, positioning accuracy (AP_P) is calculated.

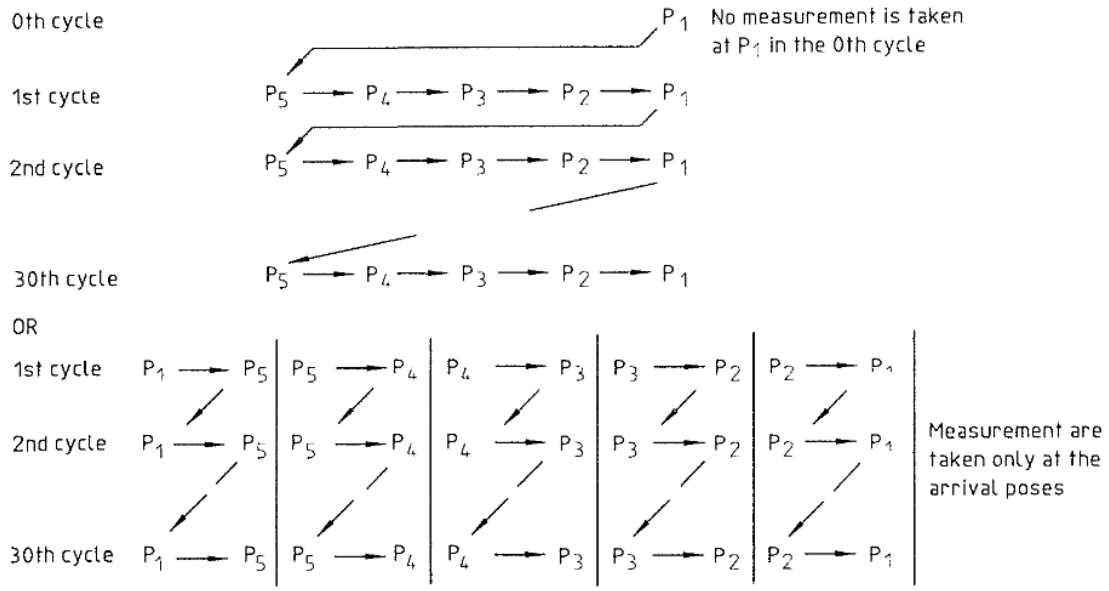


Figure 4.3: Illustration of possible cycles [9].

4.1.3 Pose repeatability

Pose repeatability expresses the closeness of agreement between the attained poses after n repeat visits to the same command pose in the same direction [9]. Its mathematical definition is expressed as follows:

$$RP = \bar{I} + 3S \tag{4.2}$$

with

$$\bar{I} = \frac{1}{n}\sum_{j=1}^{n} I_j$$
$$I_j = \sqrt{(x_j - \bar{x})^2 + (y_j - \bar{y})^2 + (z_j - \bar{z})^2}$$

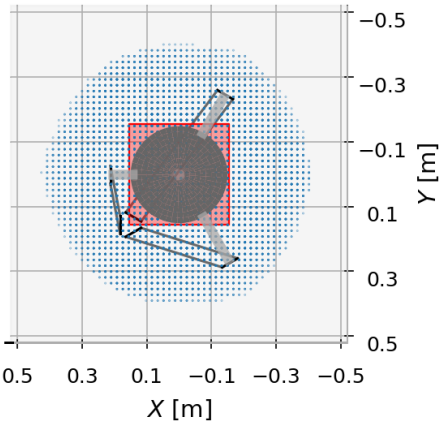
with $\bar{x}$, $\bar{y}$, $\bar{z}$ and x_j, y_j, z_j defined as in 4.1.3.

$$S = \sqrt{\frac{\sum_{j=1}^{n}\left(I_j - \bar{I}\right)^2}{n-1}} \tag{4.3}$$

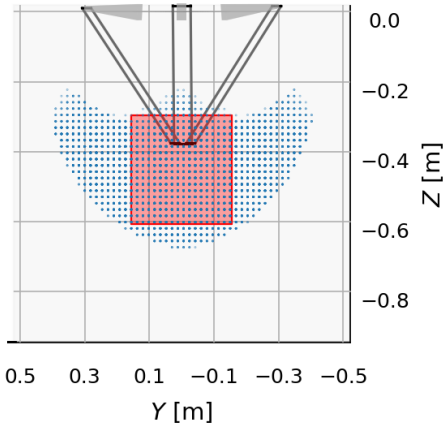
The test conditions and the procedure are the same as in 4.1.3.

4.2 Implementation

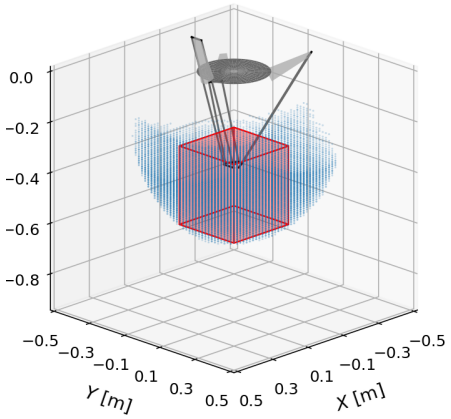
Following the procedure detailed in Section 4.1, one must start by locating a single cube in the working space. To do this, the reachable workspace is computed as in [3]. First, a 3D grid of points is generated within the potential workspace, ensuring all possible positions are considered. The inverse kinematics problem is then solved for each point to calculate the joint angles required to position the end effector. These angles are checked against the robot's mechanical limits, and points with valid angles are considered reachable.
Once the reachable workspace is determined, the next step is to find the largest cube that fits within this workspace. This involves identifying the maximum side length that does not exceed the workspace boundaries by iterating through possible cube positions and sizes, ensuring each vertex remains within the reachable points. The largest cube meeting these criteria is selected as the optimal fit.
As shown in Figure 4.4, the cube that meets the criterion stated in Section 4.1 is centered at $(0, 0, -0.475)$ and has side lengths of $0.25[m]$.



(a) Z-view of the working space.



(b) X-view of the working space.



(c) Global view of the working space.

Figure 4.4: Cube within the working space of the Delta robot.

The plane chosen is the C_4 - C_1 - C_6 - C_7 one, as defined in 4.1.1, and the points P_1 to P_5 are selected once again according to the criterion stated in Section 4.1. In the Delta Robot's workspace, those points are situated as follows.

	x	y	z
P_1	0	0	-0.475
P_2	0.125	0.125	-0.35
P_3	0.125	-0.125	-0.35
P_4	-0.125	-0.125	-0.6
P_5	-0.125	0.125	-0.6

Table 4.2: Coordinates of the poses P_1 to P_5.

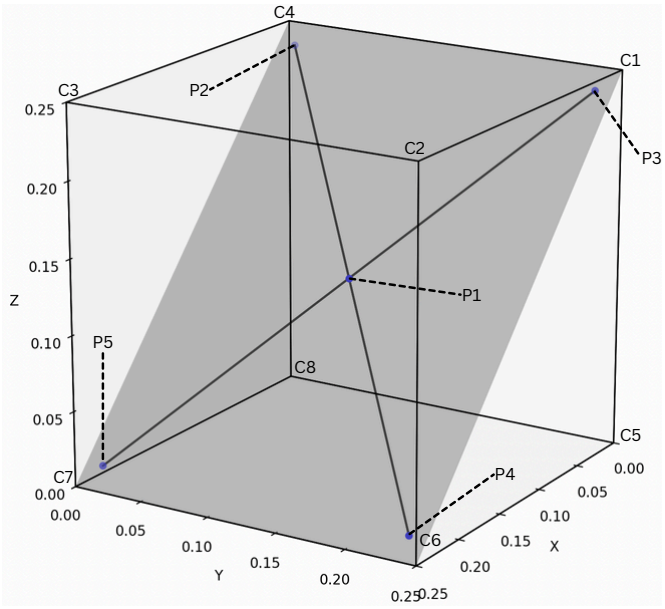


Figure 4.5: Poses to be used on the Delta's cube.

Once the poses were defined, the Delta Robot was commanded to move in a linear trajectory described by the equation:

$$\boldsymbol{X}(t) = \boldsymbol{X}_s + (\boldsymbol{X}_e - \boldsymbol{X}_s) \cdot \frac{s(t)}{x_f} \tag{4.4}$$

where $\boldsymbol{X}_s$ and $\boldsymbol{X}_e$ are the starting position and end position vectors, respectively.

The motion law for the linear trajectory uses a triangular velocity profile, the fastest way to travel from point A to point B. It involves accelerating to a maximum velocity and then decelerating immediately, with equal time and distance for both acceleration and deceleration phases. Here, x_f is the path length and T is the trajectory time:

1. Polynomial function with positive jerk for $t \in \left[0; \frac{T}{2}\right]$.

$$s(t) = 2x_f \frac{t^2}{T^2} \tag{4.5}$$

2. Polynomial function with negative jerk for $t \in \left[\frac{T}{2}; T\right]$.

$$s(t) = x_f \left[-1 + 4\frac{t}{T} - 2\frac{t^2}{T^2}\right] \tag{4.6}$$

The Delta robot was commanded to move, as stated, in a linear trajectory, taking measurements at each arrival poses. This process was repeated 30 times for each trajectory segment: from P_1 to P_5, P_5 to P_4, P_4 to P_3, P_3 to P_2, and P_2 to P_1, at the different velocities mentioned in Section 4.1, namely 100% and 50% of the rated velocity.

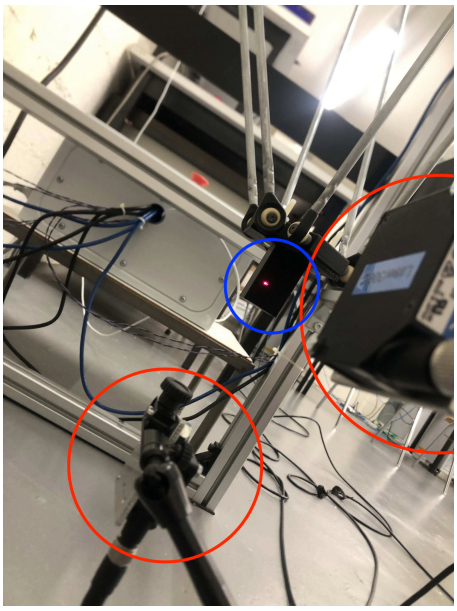
Since the robot's purpose is to serve as a demonstrator, the rated velocity is the one used during the demonstration. The demonstration aims to show high speed and excellent precision while ensuring that the viewer can visually understand the movement. Although the maximum acceleration is $11.245[g]$, an acceleration of $7.8[g]$ was chosen for clarity. With a triangular velocity profile, the maximum velocity can be expressed as follows:

$$v_{\max} = 2\frac{x_f}{T} \tag{4.7}$$

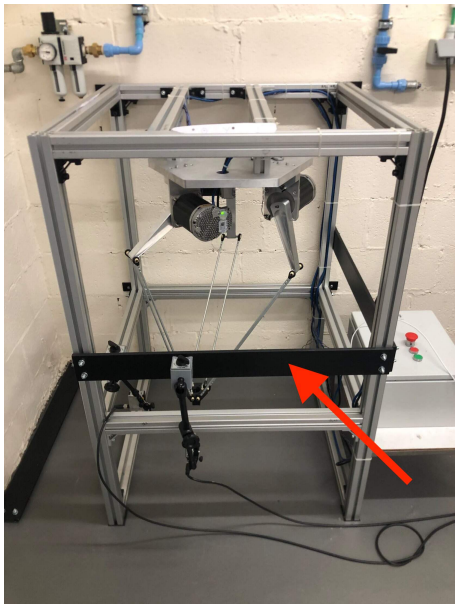
Knowing the distance between two poses from Table 4.2 and the time it took from the tests developed in Section 4.3, one can find that the maximum velocity is equal to $v_{\max,\ 100\%} = 2.5[m/s]$.

As there is no direct control over the speed, imposing the halved rated velocity to the end effector is more tricky. It required trial and error to find that with an acceleration of $4.6[g]$, the end effector reaches a maximum velocity of $v_{\max,\ 50\%} = 1.3[m/s]$.

4.3 Experimental Setup and Data Acquisition



(a) Laser projection on the end effector.



(b) Photo taken parallel to the Y-plane.

(c) Attachment device used to reduce vibration.

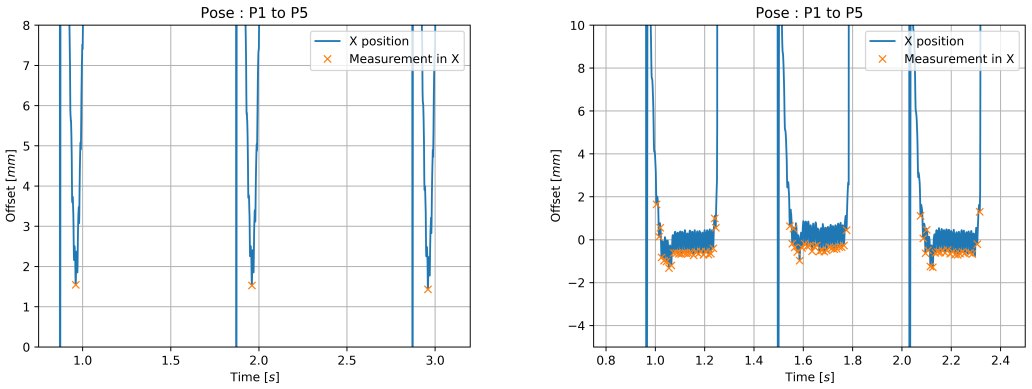
Figure 4.6: Laser setup for data acquisition.

Figure F.1 illustrates the experimental setup utilized to perform the measurements. The setup includes three lasers (Appendix D), each oriented along one of the principal spatial directions: X, Y, and Z. These lasers are mounted using a combination of electromagnets and adjustable metallic planks, allowing quick and easy installation while ensuring precision and stability during the experiments.

- Figure 4.6a provides a closer look on how the laser, circled in red, strikes the end effector, circled in blue, in order to take the measurement.
- Figure 4.6b offers a more global view of the setup, where the metallic planks, fixated with bolts and screws, can be seen (red arrow).
- Finally, Figure 4.6c presents the fixation system. The robot was fixed to the wall in order to limit vibration, hence increasing precision.

The data was then collected thanks to Acquimem, an app developed by Mr Bietlot, and two approaches were chosen to take the measurements. The first approach, illustrated in Figure 4.7a, involved a continuous back-and-forth motion between two measurement points. A Python filter was then applied to extract the local minima of the trajectory, corresponding to the poses to be measured. By knowing the position of each pose, it was straightforward to search for the local minima in the expected region of arrival. The Python list was iterated, and the local minima were identified as the pose position was approached.

The second approach, represented in Figure 4.7b, involved making a quick stop upon arriving at each measurement point. The same filter was applied in this approach, but with more data collected at the final position. As we obtained discrete points from the previously mentioned position, the mean was calculated to provide a single measurement point for each arrival.



(a) Without stopping upon arriving.

(b) With a quick stop upon arriving.

Figure 4.7: Position of the end effector upon arrival in P_5.

Unfortunately, the second approach shows that the measurement taken when the robot stops at the pose is not accurate, as seen in the zoomed view in Figure 4.8. The robot exhibits oscillations when stopped because the motors continuously correct the position of the end effector due to small vibrations in the room. These vibrations cause an offset of approximately half a millimeter. Therefore, taking only the minima of these oscillations would not be meaningful. Another filter or method should be used.

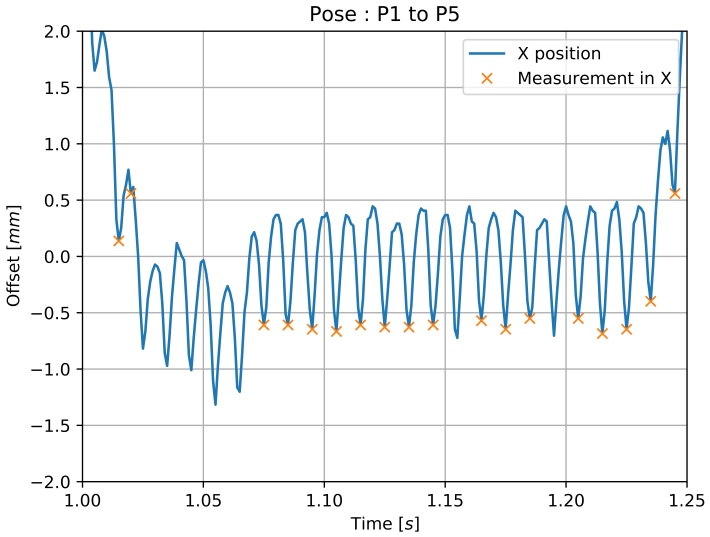


Figure 4.8: Position of the end effector upon arrival in P_5 in the X-direction.

One quick easy fix is to use the local maxima as well, as shown in Figure 4.9, in the mean's calculation.

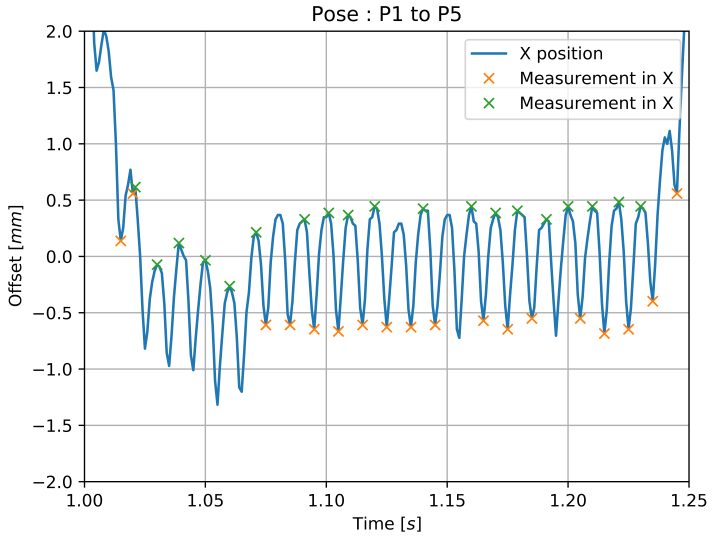


Figure 4.9: Position of the end effector upon arrival in P_5 in the X-direction.

Finally, all points during the level can be taken, as shown in Figure 4.10. Even though this method is harder to implement, it should be more accurate.

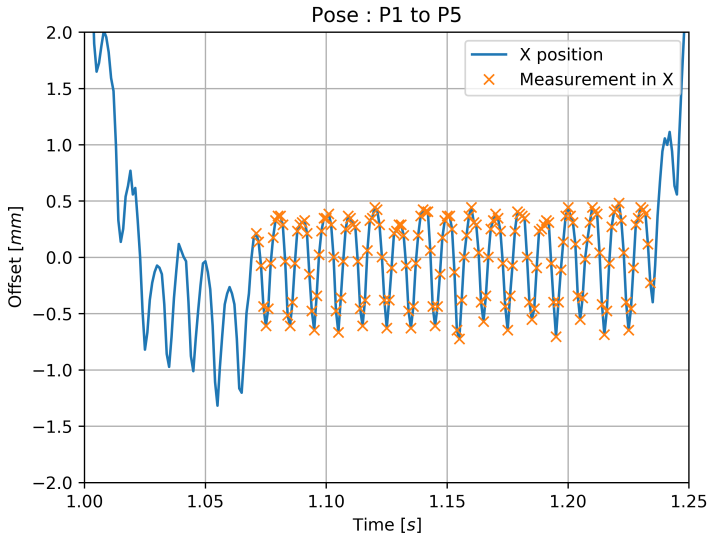


Figure 4.10: Position of the end effector upon arrival in P_5 in the X-direction.

4.4 Results and analysis

In this section, the different methods and filters will be compared and further analysed in order to state whether or not the Delta robot is precise.

4.4.1 Pose accuracy

Using (4.1), the pose accuracy can be calculated for each of the methods. Below are the results for P_5; the complete precision test results are detailed in Appendix C.

1. Without stopping upon arriving: $AP_P = 1.9mm$
2. With a quick stop upon arriving:
   (a) Local minima only: $AP_P = 0.62mm$
   (b) All extrema: $AP_P = 0.096mm$
   (c) All points: $AP_P = 0.12mm$

4.4.2 Pose repeatability

As for the pose accuracy and using (4.2), details follows as:

1. Without stopping upon arriving: $RP = 0.085mm$
2. With a quick stop upon arriving:
   (a) Local minima only: $RP = 1.4mm$
   (b) All extrema: $RP = 0.19mm$
   (c) All points: $RP = 0.27mm$

As stated previously, taking into account all the points while the robot is stopped should provide the most accurate measurement. However, this method is quite tedious to implement because it cannot be automated due to the varying arrival times. Therefore, the method of using extrema will be preferred, as it is much easier to implement and only minimally sacrifices accuracy. As shown below, the error is around 3%:

$$\epsilon_{AP} = \frac{|0.096 - 0.12|}{0.12} = 0.2 \tag{4.8}$$

$$\epsilon_{RP} = \frac{|0.19 - 0.27|}{0.27} = 0.3 \tag{4.9}$$

These calculations illustrate that while the extrema method introduces some error, it remains within an acceptable range for practical applications.

Furthermore, the level method is favored over the non-level one, even though it doesn't highlight the robot's repeatability as effectively. Since the measurement method using lasers is discrete, having a slightly longer measurement period helps to mitigate the effects of discrete timing.

One can ask itself why the pose repeatability is better without stopping time than the one with. This is due to the oscillation when the robot is stopped caused by small vibration in the room.

The detailed data for these measurements can be found in Appendix C. The conclusion from these tests is that the Delta robot demonstrates exceptional accuracy and repeatability, exceeding the desired performance stated in Chapter 2, making it an ideal demonstrator.

Chapter 5

Flexibility analysis

5.1 Introduction

The aim of this study is to determine if the distal arms of the Delta robot exhibit flexibility, which could impact its precision. Identifying and quantifying this flexibility is crucial for maintaining accuracy, especially in high-speed and high-acceleration scenarios. Given the high accuracy demonstrated in Chapter 4, it can be hypothesized that any flexibility will likely be minimal.

The focus on the flexibility of the distal arms arises from a complete FEM simulation of the entire kinematic chain of the Delta robot, conducted during the previous master's thesis [3], which shows that these arms endure the most deformation.

Directly placing sensors on a robot operating at high speeds and accelerations poses challenges, as the forces and vibrations could compromise sensor placement and data accuracy. Therefore, an alternative approach is needed to measure the flexibility of the distal arms without interfering with the robot's performance.

Moreover, since this robot is designed as a demonstrator, capturing any flexibility through camera images would be particularly compelling. To address this, a method using high-speed camera technology is proposed to analyze the flexibility of the robot arms.

The following sections outline the setup, describe the method, validate its accuracy, and finally apply it to the Delta's distal arms.

5.2 Materials and methods

In this section, we describe the experimental setup and method used to analyze the flexibility of the Delta robot arms.

5.2.1 Test specimens

The flexibility analysis of the Delta robot arms involves two primary test specimens: a wooden flower stake (shown in Figure 5.2a), that will be further denoted as test beam for clarity, used to validate the measurement method, and the actual distal arms of the robot.

The significant flexibility of the test beam allows for clear observation and measurement of deflection, helping to assess the accuracy and reliability of the proposed method before it is applied to the actual robot arms.

The distal arms, KDGM-06-A-SR-J from IGUS, are made of galvanised steel and are 50$[cm]$ long.

5.2.2 Test set-ups and measurements

To present and validate the proposed method, an experimental setup distinct from the robot itself is employed. This alternative setup is more favorable to validate the method, ensuring that the measurements and analysis are accurate and reliable. Once the method is confirmed using this setup, it will then be applied to measurements taken directly from the robot.

Test set-ups

Let's first examine the common equipment used in both setups.

Both setups utilize the Photron Mini UX100 type 800K-M-32G high-speed camera (Appendix E), shown in Figure 5.1. This camera is widely used in general scientific and industrial applications. It captures black-and-white images with a resolution of 1.3 megapixels (1280 x 1024 pixels) and supports frame rates of up to 4,000 fps.

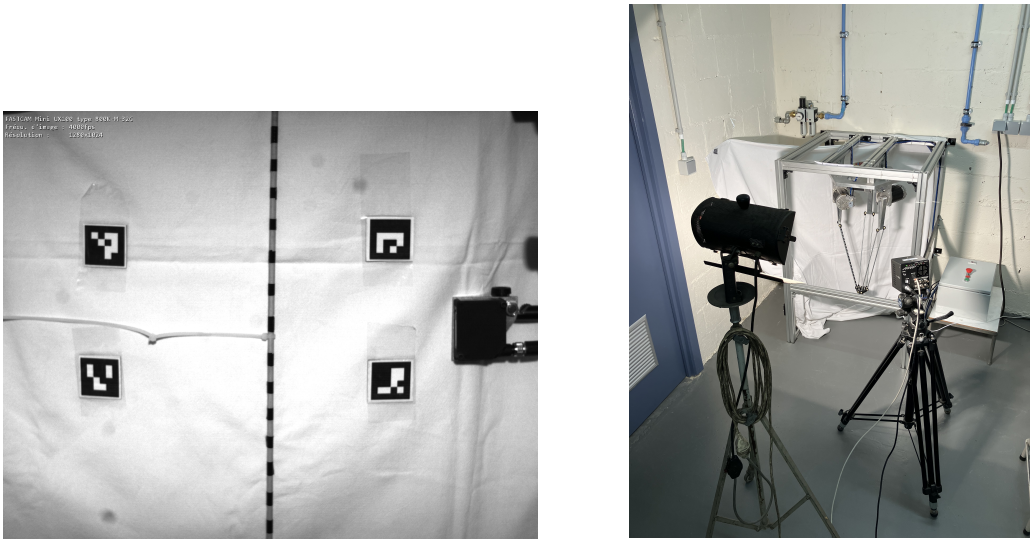
Figure 5.1: Fast camera Photron Mini UX100 type 800K-M-32G.

For high-speed imaging, proper lighting is required. A large light source (shown in Figure 5.2 (b)) is used to ensure that the scene is well-illuminated. High-speed cameras require intense lighting because they operate at very high frame rates, which means that each frame has a very short exposure time. Without sufficient light, the captured images would be too dark and lack the necessary detail for accurate analysis. The large light source helps to eliminate shadows and provides uniform illumination, ensuring that every important element of the scene and other features on the beams are clearly visible in every frame.

To further facilitate the image processing process, a white sheet is used to create a more homogeneous background. This helps in enhancing the contrast and visibility of the markers and other relevant features, making it easier to process and analyze the images accurately.

Additionally, four 4x4 ArUco markers (with IDs: 0,1,2 and 3) are placed at each corner of a square surrounding the beams (see Figure 5.2b). They will be used to facilitate precise tracking and measurement.

Finally, let's see what differentiate the two set-ups. The set-up used to validate the method includes a laser (its data-sheet is available in Appendix D), that we can see in Figure 5.2b. Moreover, while the first setup uses a test beam attached at both ends, the second setup uses the actual distal arms of the robot, allowing them to move in their operational manner.



(a) (b)

Figure 5.2: Test set-ups : (a) validation set-up (b) real distal arms set-up.

Measurements

For the first test, which aimed to validate the method, a force was manually applied to the center of the test beam by pulling on a rope attached to its midpoint. The exact magnitude of the force was not measured, as the primary objective was to induce bending (deflection) in the beam.

In the robot's setup, a back-and-forth linear trajectory at the robot's, detailed further, was utilized.

The actual deflection/displacement was measured using the previously mentioned laser displacement sensor. These measurements will be compared with the deflection/displacement data obtained through image processing.

5.3 Image processing

The proposed image processing method for measuring distal arm deflection involves the following main steps, as illustrated in Figure 5.3, all of which are detailed in the following section.

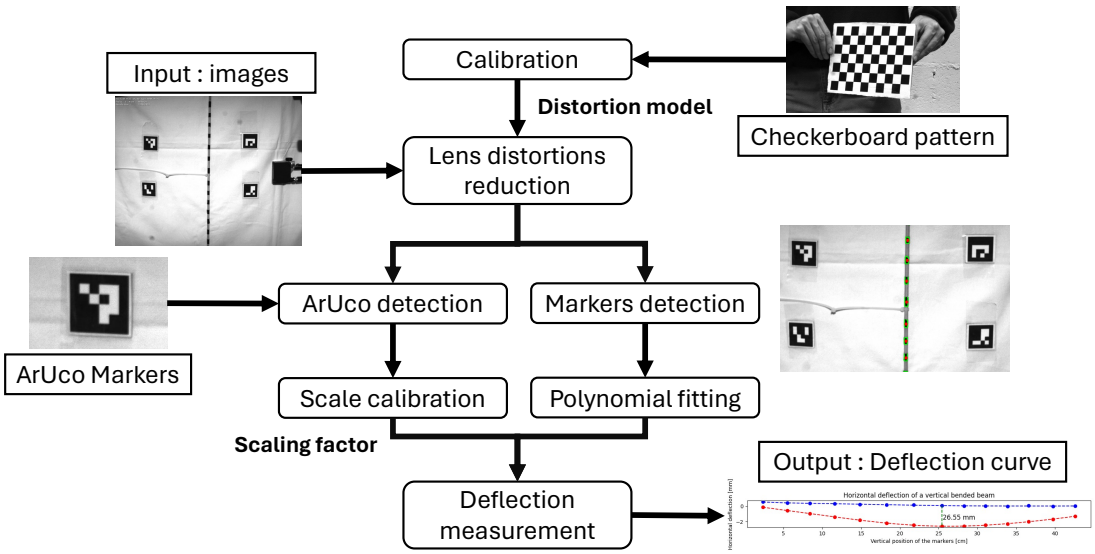


Figure 5.3: Main steps of the developed method.

5.3.1 Camera calibration

An image can be represented as a function, such as $I(x, y)$, where x and y are the coordinates of a pixel, and the value of $I(x, y)$ indicates the gray level of the image at that specific point. As explained in [50], processing these images allows for the reconstruction of a 3D scene, enabling the detection and measurement of real-world objects. However, the accuracy of captured images is often compromised by distortions introduced by camera lenses and sensors. To correct these distortions, it is essential to perform camera calibration, which involves determining both the intrinsic and extrinsic parameters of the camera. This calibration is crucial for various applications, including correcting lens distortion and measuring object dimensions in real-world units.

The main goal of camera calibration is to accurately determine the intrinsic and extrinsic parameters of the camera. Intrinsic parameters encompass the focal length, optical center, and skew coefficient, while extrinsic parameters define the camera's position and orientation relative to the world. Among the different calibration techniques, a notable approach is the one proposed by Zhengyou Zhang in 1998 [51], recognized for its flexibility and precision. This technique employs a planar calibration pattern viewed from multiple orientations to accurately determine the camera's parameters.

The theoretical foundation for camera calibration is the *pinhole camera model*[1] (other models exist [52]), which idealizes a camera as a single point, the "pinhole", that projects 3D points onto a 2D image plane. The one shown in Figure 5.4 will be based on three elementary transformations combined together. The following development comes from [53].

In this model, a 3D point M in the scene, defined by its coordinates in the world frame R_W, (X_W, Y_W, Z_W), is represented in the camera's local frame R_C, (X_C, Y_C, Z_C).

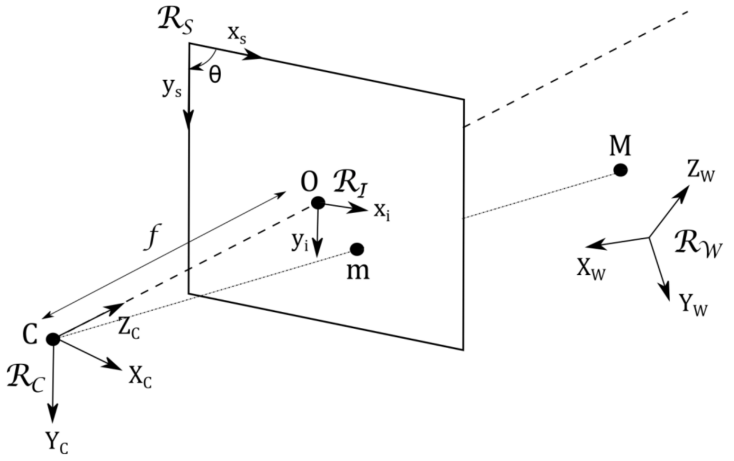


Figure 5.4: Projection of a 3D point onto an image with a pinhole camera model, from [53].

The first elementary transformation that we consider is the conversion between the homogeneous world coordinates $\{X_W\}$=$\{X_W, Y_W, Z_W, 1\}$ and the local coordinates $\{X_C\}$=$\{X_C, Y_C, Z_C, 1\}$. It is represented by a 4×4 matrix denoted T and can be decomposed into a rotation matrix R (parameterized by three rotation angles) and a translation vector t (defined by three components). This gives six parameters that one call the *extrinsic parameters*.

The second transformation aims to project the 3D point M on the image plane at a 2D point m, shown on Figure 5.4. This step only involves the focal length f of the camera and a scale factor s.

The third transformation is used to represent the homogeneous 2D coordinates $\{x_s\} = \{x_s, y_s, 1\}^T$ of the projected point m within the sensor frame R_S. This transformation takes into account the sensor's specific characteristics:

[1]This model is preferred for its simplicity and effectiveness in projecting 3D points onto a 2D image plane.

- the skew angle between the horizontal and vertical axes of the sensor, assumed to be 90°;
- the position of the optical center;
- the physical size of the pixel in both directions.

One can combine and express the two last transformations through a projection matrix K that is defined by the four *intrinsic parameters* of the camera (the horizontal focal lengths f_x and vertical f_y in pixels and the position (c_x, c_y) in pixels of the optical center in the image).

Finally, one can describe the camera with the projection matrix M that is composed of a matrix of extrinsic parameters T and a matrix of intrinsic parameters K, in the following way :

$$s \cdot x_s = [K][T]X_W = [M]X_W$$

The objective of camera calibration is to determine the projection matrix M. The previous model doesn't take distortions into account as an ideal pinhole camera does not have a lens [54], so we have to complete it by introducing the two major kinds of distortion : radial and tangential distortion. With radial distortion that causes straight lines to appear curved, and tangential distortion, which happens when the image plane and the lens are not parallel, making some areas in the image look nearer than they should be [55].

Before starting the calibration process, a specific pattern, such as the checkerboard in Figure 5.5, must be attached to a sufficeintly flat surface. This pattern includes points with precisely known coordinates on the plane.

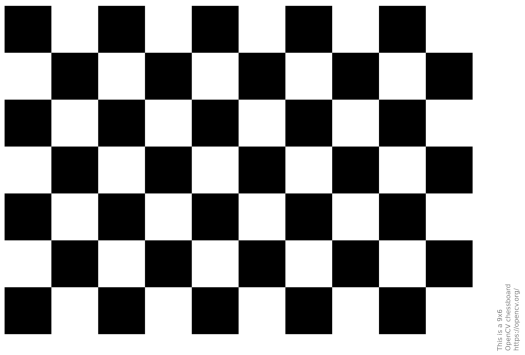


Figure 5.5: OpenCV calibration checkerboard pattern image.

Calibration starts by capturing multiple images of the pattern from various orientations and positions. This variety of views is important for accurately determining

the camera parameters. The pattern can be moved (as is done here) or, alternatively, the camera can be moved.

Now that multiple images have been captured, the next goal is to solve for the camera parameters. The first step is *homography estimation*. Each image of the checkerboard provides a set of corresponding 2D image points and known 3D object points. These points are used to compute the homography H for each image using Direct Linear Transformation (DLT), based on the pinhole camera model.

The second step involves optimization. After computing the homographies for each image, the intrinsic parameters and lens distortion coefficients are refined. This is done by minimizing the re-projection error, which is the sum of the squared differences between the observed image points and the projected points from the 3D model. The optimization problem can be formulated as follows:

$$\min_{K,\mathbf{R},\mathbf{t},k_1,k_2,p_1,p_2} \sum_{i=1}^{N} \|\mathbf{x}_i - \text{project}(K, \mathbf{R}, \mathbf{t}, k, p, \mathbf{X}_i)\|^2$$

where:

- K represents the intrinsic camera matrix including the focal lengths and the principal point.
- $\mathbf{R}$ and $\mathbf{t}$ represent the rotation and translation vectors describing the camera's pose.
- k_1, k_2, p_1, p_2 are the coefficients for radial and tangential distortions.
- $\mathbf{x}_i$ are the observed 2D image points.
- $\mathbf{X}_i$ are the corresponding 3D world points.
- project is a function that applies the camera model, including distortion corrections, to project the 3D points onto the 2D image plane.

The `project` function computes the position of the 3D point in the camera's image plane, taking into account both the geometric projection and the distortion induced by the camera lens. This optimization process uses nonlinear least squares to find the parameters that best fit the observed data.

Figure 5.6: Camera calibrated image example, original (left) undistorted (right).

Figure 5.7: OpenCv checkerboard calibration pattern.

After calibration, the parameters are used to correct image distortions and for precise dimension measurements. One typically assesses the effectiveness of the calibration using the reduction in re-projection error. The latter one represents the difference between the observed image points and the projected 3D points onto the image plane [54]. As shown in Figure 5.8, the calibration performed very well as the mean error is closed to zero, even if it will never become exactly zero [56].

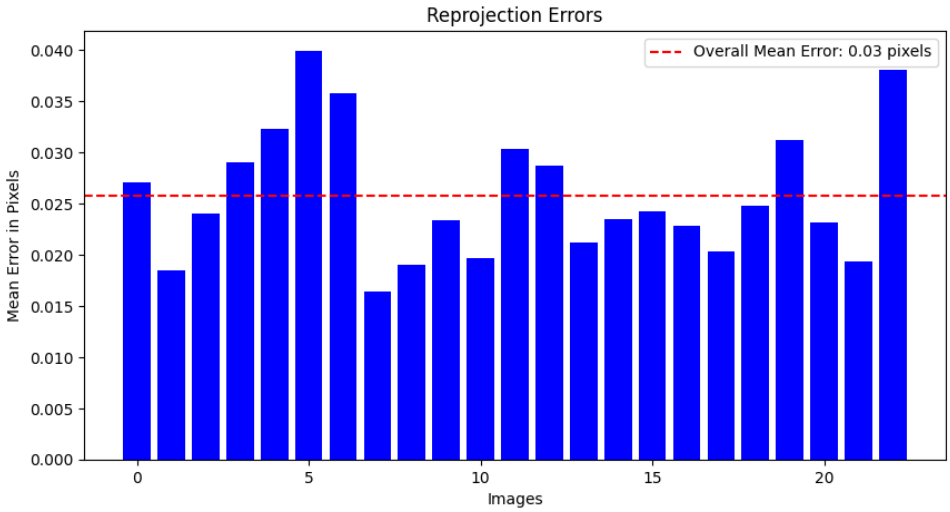


Figure 5.8: Re-projection errors after camera calibration.

5.3.2 Markers detection and tracking

The proposed method employs tracking to achieve its goal. To accurately measure distances in images, it's important to account for lens distortion. Camera calibration involves determining the camera's intrinsic parameters and distortion coefficients, as explained in the previous section. Using these parameters, one undistorts the images to correct for lens distortions, ensuring more accurate measurements.

Then, the process involves detecting the four markers of the setup. ArUco markers were chosen over other types due to their high detection accuracy and robustness against varying lighting conditions and partial occlusions. Additionally, their simple design allows for efficient and reliable identification using standard computer vision techniques.

Their detection process is composed of two main phases [57, 58]. First, the image is analyzed to identify square shapes that could be markers. This process begins with adaptive thresholding to segment the markers. This technique converts a grayscale image into a binary image by calculating a threshold for smaller regions (windows) of the image rather than using a single global threshold. This allows for better handling of varying lighting conditions within the image. The local threshold can be determined using methods such as the mean, Gaussian weighted sum, or median of pixel values within each region.

Then, one extracts contours from the thresholded image. Contours that are not convex or do not approximate a square shape are discarded, and additional filtering

removes those that are too small, too large, or too close to each other. The used algorithm is based on the Suzuki and Abe algorithm [59].

Next, to validate potential markers, their internal codification is analyzed. A perspective transformation is applied to obtain the marker in its canonical form. This canonical image is then thresholded using Otsu's method to differentiate white and black bits [60]. The image is divided into cells based on the marker and border size, and the number of black or white pixels in each cell determines the bit color. Finally, the bits are examined to ascertain if the marker belongs to a specific dictionary.

As shown in Figure 5.9, the ArUco markers are successfully detected following this procedure.

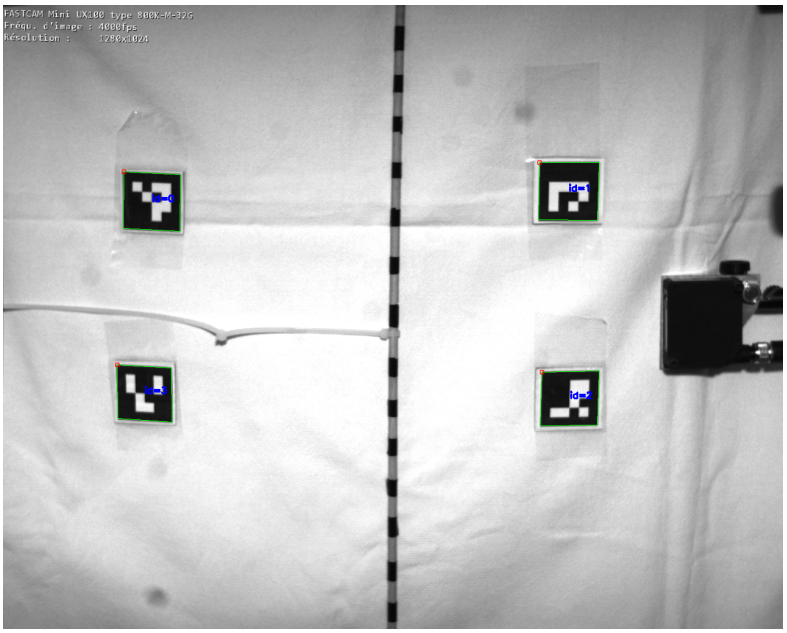


Figure 5.9: Detected ArUco markers on method validation set-up.

Once these markers are detected, the next task is to compute image scaling factors. These are needed to translate the pixel measurements captured by the camera into real-world units. By knowing the precise distances between the different ArUco markers, one can establish these scaling factors, ensuring that the measurements of distances within the image accurately reflect the real physical dimensions.

The next step consist in tracking the robot's arm motion. To do so one have to detect the black markers on one of the robot's distal arms. This process, though simple in concept, requires several key steps to ensure accurate and efficient marker

detection. To simplify the method, a user intervention is required to initiate the detection process. Specifically, a user must manually click on the location of the markers to help the system find them.

Once the user clicks on the image, the program retrieves the intensity value at the clicked point. This value is then used to set the binary threshold. This technique converts a grayscale image into a binary image where each pixel is either 0 (black) or 255 (white). By defining a threshold value, pixels in the grayscale image with intensity values above this threshold are set to white, while those below are set to black. In the resulting binary image, areas corresponding to the markers (usually darker regions) are clearly separated from their surroundings, as shown in Figure 5.10, facilitating further analysis.

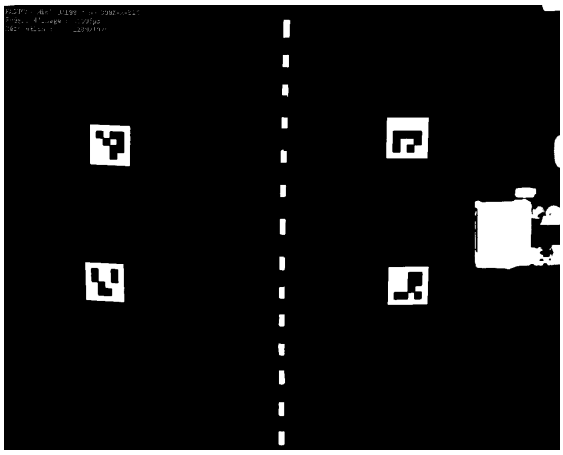
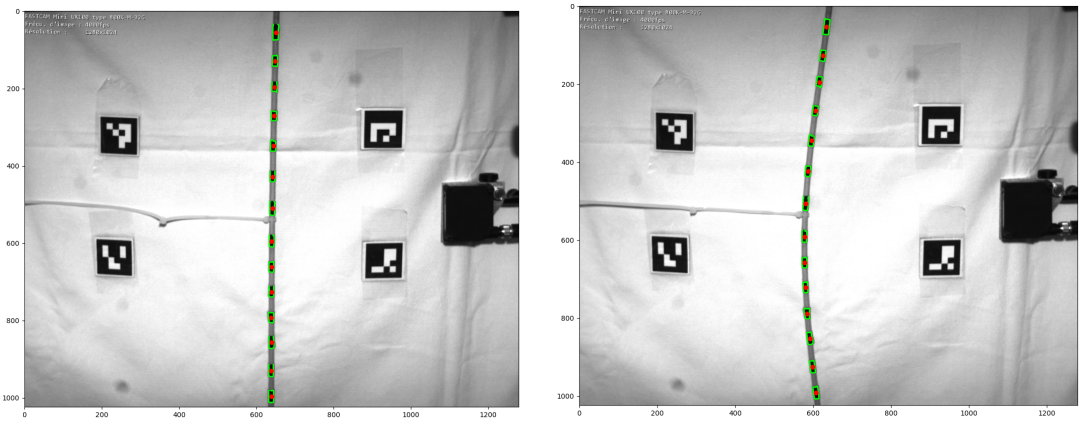


Figure 5.10: Image of the validation set-up after binary thresholding.

Once the binary image is prepared, contour detection is employed to locate the boundaries of the markers. Contours are continuous curves that connect all the points along a boundary sharing the same color or intensity. This technique identifies and delineates these boundaries within the binary image, representing the shapes and positions of the markers, as previously explained for ArUco markers detection.

By scanning the binary image, the system retrieves the contours, producing a list of points that outline each detected object. Analyzing these contours allows the system to accurately identify the markers' positions and shapes. The system checks if the clicked point falls within any detected contour. If a contour is found, its properties, such as the bounding box and center, are calculated. Figure 5.11 shows the contours found by the program, outlining them in green and marking their centers in red.



(a) (b)

Figure 5.11: Markers detection for motion tracking. (a) Initial position of the test beam and (b) final position of the test beam.

5.4 Data Analysis and Method Validation

5.4.1 Deflection Measurement and Analysis

After one computes the centers of the markers, the next step is to analyze the test beam's deflection by fitting polynomial curves to the markers centers.

First, one converts the horizontal and vertical positions of the markers from pixel units to real-world units (centimeters) using the scaling factor previously computed.

Next, one focuses on fitting polynomial curves to the data. A polynomial function $P(y)$ is defined to model the relationship between the vertical positions of the markers (y) and their horizontal deflections (x). This polynomial function is expressed as:

$$P(y) = c_0 + c_1 y + c_2 y^2 + c_3 y^3 + c_4 y^4 + c_5 y^5 + c_6 y^6 + c_7 y^7$$

where $c_0, c_1, c_2, c_3, c_4, c_5, c_6$, and c_7 are the coefficients to be determined. A polynomial of degree 7 has been chosen as it provides, as one will see later, a very accurate model for the complex deflection patterns of the beam, which may not be adequately captured by lower-degree polynomials (it has been tested and returned less precise results).

Once the polynomials are fitted, they are used to estimate the beam's initial and final deflection profiles. By comparing these two profiles at corresponding vertical positions, one can calculate the horizontal deflections between the initial and final states of the beam.

5.4.2 Validation of the Method

To ensure the accuracy and reliability of the proposed method for analyzing the flexibility of the Delta robot distal arms, the use of a validation process is required. It involves comparing the results obtained from the developed method with established techniques to confirm its reliability. To do so, one uses laser displacement sensor as a reference.

The output of the laser is shown in Figure 5.12. The blue curve, raw data from the laser, shows some noise in the data. This noise can stem from various sources such as sensor imperfections, or environmental vibrations.

In order to get a more precise value one applies a filter to the raw data. The Savitzky-Golay filter was chosen for this application. The orange curve represent the data after applying the latter one. Unlike other smoothing filters, such as moving average or Gaussian filters, the Savitzky-Golay filter is designed to preserve the higher-order moments of the signal [61]. Using it one keeps essential features from the signal such as peaks and edges. The latter ones are important for accurate deflection measurement. It works by fitting successive subsets of the data with a low-degree polynomial using the method of least squares. As illustrated, it does effectively reduce the noise without distorting the original signal.

Finally, one take the maximum value of the filtered curve and get a deflection of $26.94[mm]$.

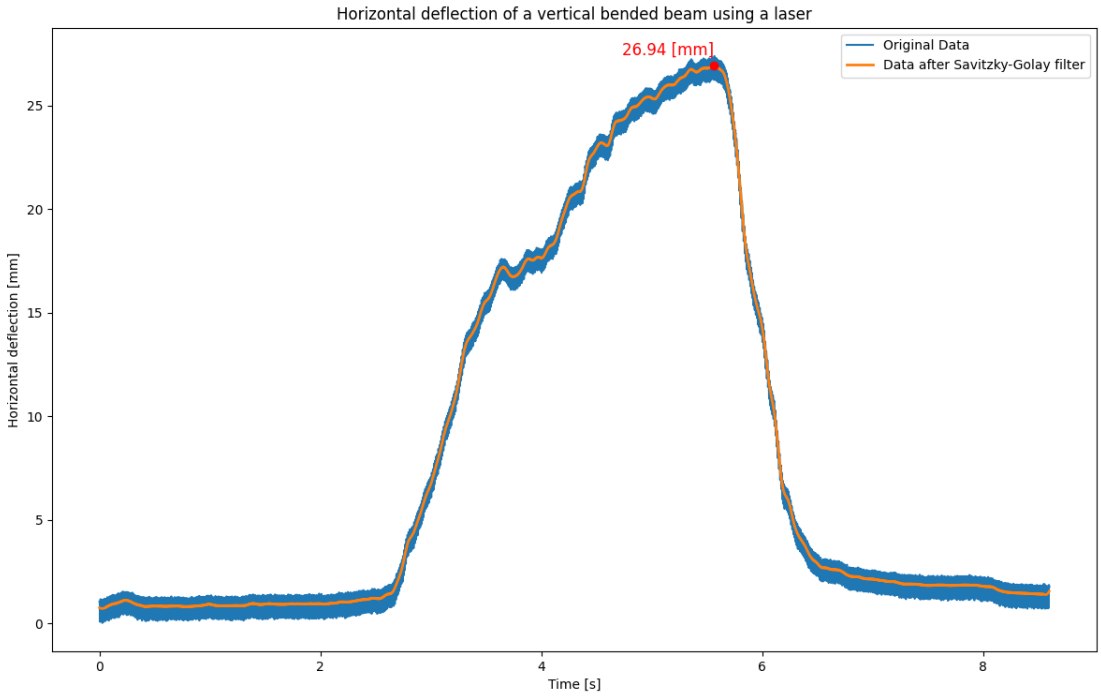


Figure 5.12: Graph of the horizontal deflection of the test beam at under applied force at its midpoint, measured by a laser. The blue curve shows the raw laser data, while the orange curve represents the data after applying a filter to reduce noise.

After applying the previously developed method to the Figures 5.11 taken on the validation set-up, one gets the graph shown in Figure 5.13. The resulting value for the deflection of the beam at the same position where the laser took its measure equal to 26.55$[mm]$. This implies a difference of about 1.447% from the reference method using a laser. Which means that the developed method as a precision up to few tenth of millimeters.

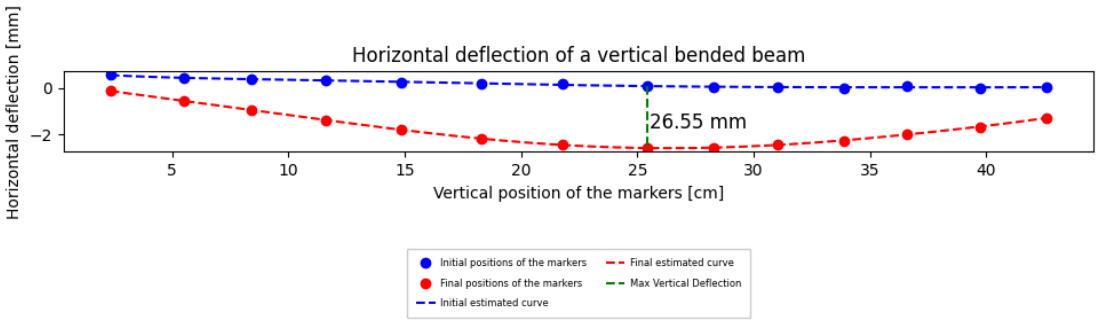


Figure 5.13: Graph of the horizontal deflection of the test beam under applied force at its midpoint, measured by the implemented vision-based method.

Based on these results, one can conclude that the developed method using a high-speed camera is effective for detecting bending in the robot's distal arms. Moreover, it can accurately quantify deflections in the range of a few tenth of millimeters.

5.5 Specific application and result

The validated method is applied to the Delta robot, which follows a back-and-forth linear trajectory (described by Equation 4.4) at its maximum acceleration to perform the measurements. To induce maximum deformation in the robot's arm, a motion law with a triangular velocity profile was applied (as described by Equations 4.5 and 4.6). This law was chosen because it produces infinite jerk (as will be discussed later in Section 6.2.2), resulting in maximum deformation.

The input image used for processing is taken at the point when the robot changes direction at the end of its linear trajectory. Following the steps detailed in Sections 5.3.1 and 5.3.2, the markers on the robot's distal arms are accurately detected, as shown in Figure 5.14.

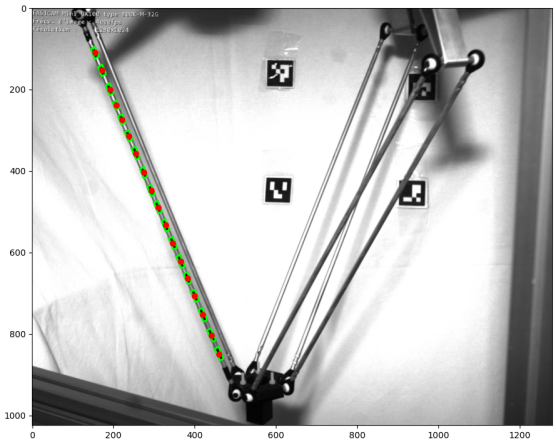


Figure 5.14: Markers detection on the robot distal arms for motion tracking.

Before assessing deflection, it's important to check if there's any bending in the robot's distal arms. To do this, a polynomial is fitted through the centers of the markers, and the quality of this fit is evaluated using the R^2 value [62]. The R^2 value, which ranges from 0 to 1, indicates how well the independent variable explains the variance in the dependent variable. An R^2 of 1 signifies a perfect fit, while an R^2 of 0 provides no explanation of the variation.

To evaluate bending, both polynomial and linear models are fitted to the data, and their R^2 values are calculated. A significantly higher R^2 value for the polynomial fit suggests a more complex relationship, while similar R^2 values for both fits indicate that the relationship is approximately linear.

In this case, both models returned R^2 values of 1, indicating that both the polynomial and linear fits perfectly describe the data. This implies that the deflection follows a straight line, which implies no bending in the distal arms.

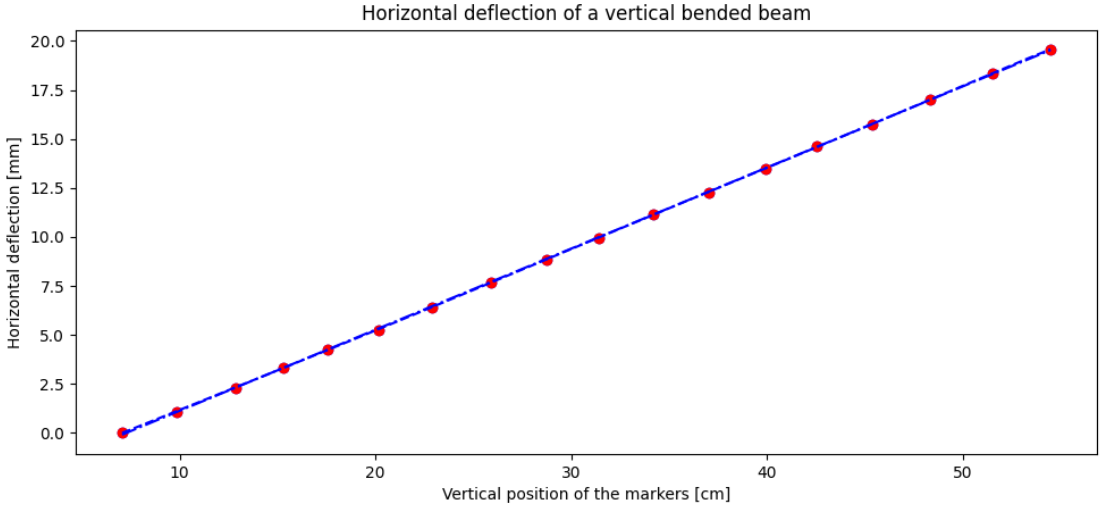


Figure 5.15: Graph of the fitted polynomial through the detected markers of the Delta distal arms.

5.6 Conclusion

An high-speed camera method to assess the flexibility of the Delta robot's distal arms was effectively implemented, demonstrating high precision in detecting deflections as small as a few tenth of millimeters. When applied to the Delta robot's distal arms, it appears that they didn't bend under high-accelerating motion. Those results are consistent with the results obtained through simulation in the previous thesis on the delta [3].

If a more precise method is desired, Digital Image Correlation (DIC), which also involves using a high-speed camera, could be utilized. DIC is a technique that measures surface displacements and strains with high precision, often up to 0,01$[mm]$. It works by comparing images of an object taken before and after deformation, then calculating the displacements at each point on the surface. This method has been proven effective in various structural tests, including those conducted on larger structures like bridges [63].

Chapter 6

Pedagogical demonstrations

After confirming the Delta robot's precision and rigidity, even at high speeds, and successfully implementing a user-friendly interface, it is now time to showcase its applications and provide demonstrations. Each demonstration will highlight either a physical phenomenon or a concrete application of the parallel robot.

6.1 Demo 1: Pick and place task followed by high speed path following

This demonstration highlights our robotic system's capabilities through a two-part task combining precision manipulation and dynamic motion control. The first phase involves a "pick-and-place" operation, where the robot accurately picks up a cylindrical object (see Chapter 2, Figure 2.3) from a designated start point and places it at a target location, both clearly marked on the demonstrational plank (see Figure 6.1). This phase showcases the system's ability to handle delicate operations with spatial precision. In the second phase, the robot transitions smoothly into high-speed path-following mode, navigating a predefined trajectory with agility and control. This sequence underscores the robustness and versatility of the system's control, offering a comprehensive evaluation of its performance under varying conditions.

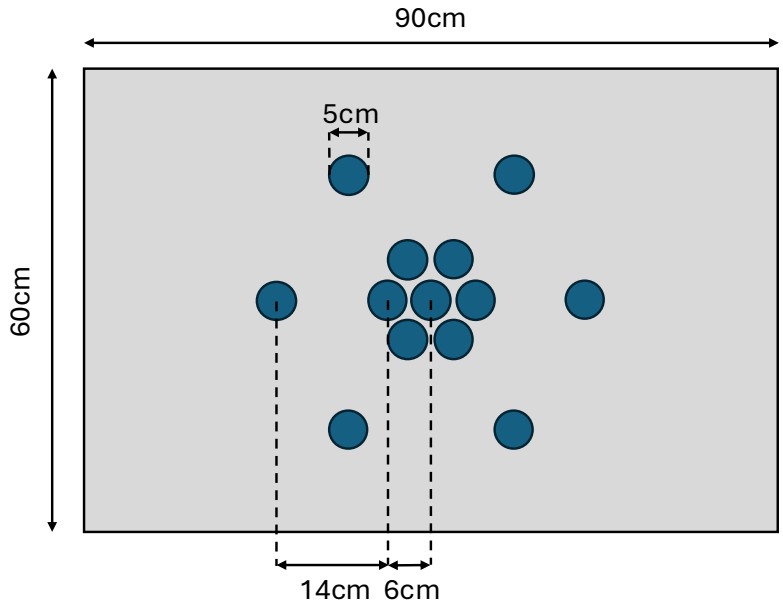


Figure 6.1: Schematic of the plank used in the demonstration.

6.1.1 Phase 1

The robot moves to its homing position, enabling the placement of the plank. To ensure consistent positioning, four screws and bolts (Figures 6.2a and 6.2b) are secured on the Bosch profile structure of the Delta robot at each corner of the plank's intended location. Once the plank is in place, the cylinder can be positioned in its designated spot on the plank.

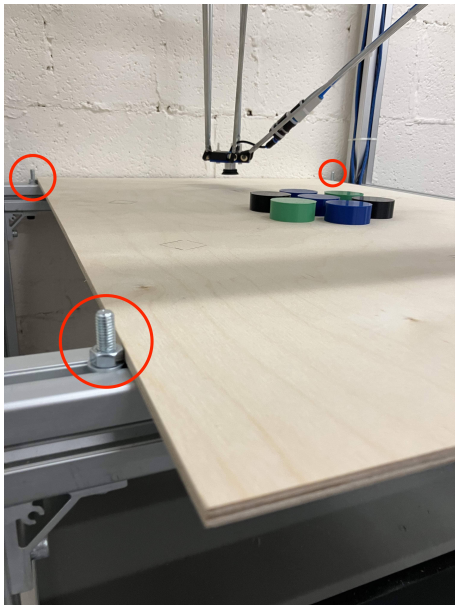
6.1.2 Phase 2

Once everything is in place, the user can press a button (Figure 6.2c), which will launch the phase two of the demonstration. The robot will start picking the cylinders and placing them on a circle, evenly spaced (see Figure 6.3). The pick-and-place trajectory follows this semi-elliptical equation, adapted from [64]:

$$\boldsymbol{X}(t) = \boldsymbol{C} + A \cdot \boldsymbol{a} \cdot \cos\left(\frac{\pi \cdot s(t)}{x_f}\right) + B \cdot \boldsymbol{b} \cdot \sin\left(\frac{\pi \cdot s(t)}{x_f}\right) \tag{6.1}$$

Here, $\boldsymbol{C} = \frac{1}{2}(\boldsymbol{X}_s + \boldsymbol{X}_e)$ represents the midpoint between the starting and end positions, $\boldsymbol{X}_s$ and $\boldsymbol{X}_e$. The parameters A and B denote half the lengths of the major and minor axes of the ellipse, while $\boldsymbol{a}$ and $\boldsymbol{b}$ are their respective unit vectors.

(a) One of the screws allowing the plank to glide in the same position every time.



(b) View of the plank constrained by the screws (circled in red).

(c) Red button allowing user inputs during the demonstration.

Figure 6.2: Elements of the demonstration.

6.1.3 Phase 3

The third phase involves maneuvering the final cylinder, initially placed at the origin $(0, 0)$, through a slalom between the previously positioned cylinders. This movement is achieved by imposing a sinusoidal oscillation onto a circular trajectory. Since a circle is a special case of an ellipse, where the major and minor axes are equal to the circle's diameter, the semi-elliptical trajectory from equation (6.1) is adapted to describe this circular path. The oscillatory circular trajectory is given by:

$$\boldsymbol{X}(t) = \boldsymbol{C} + R(s)\left[\boldsymbol{a} \cdot \cos\left(\frac{\theta \cdot s(t)}{x_f}\right) + \boldsymbol{b} \cdot \sin\left(\frac{\theta \cdot s(t)}{x_f}\right)\right] \tag{6.2}$$

Where,

$$R(s) = R + A\cos\left(\frac{n \cdot \theta \cdot s(t)}{x_f}\right) \tag{6.3}$$

In this context, $\boldsymbol{C}$ denotes the center of the circular path, R is the circle's radius, A is the oscillation amplitude, and n represents half the total number of cylinders. The vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthogonal unit vectors, defining a basis perpendicular to the circle's rotation axis. The angular frequency is given by $\theta = 2\pi f T$, where f is the frequency in revolutions per second, and T is the total duration of the trajectory.

Figure 6.3 shows in green where the cylinders are posed and in blue the path followed by the Delta robot.

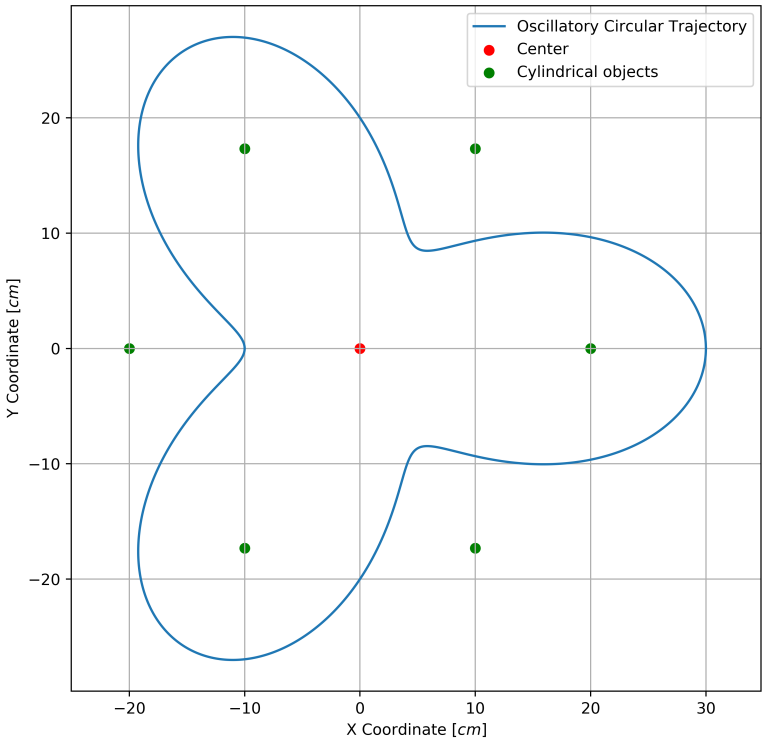


Figure 6.3: Phase 3 of the demonstration where the Delta robot slaloms (blue line) between previously placed obstacles (green dots).

6.1.4 Phase 4

After completing the slalom, the robot returns the cylinder used in the slalom to the center. It then initiates another pick-and-place sequence to reposition the evenly spaced cylinders around the centered one.

6.1.5 Phase 5

Finally, the robot returns to its homing position, allowing the user to remove the plank. Once the plank is removed, pressing the same button used in Phase 2 will shut down the motors.

6.2 Demo 2: Importance of motion law in trajectory planning

A trajectory can be considered as the combination of a path and a motion law [65]. The path describes the end-effector's motion in the task space geometrically. It is represented mathematically as a parametric curve $\boldsymbol{X}(s)$, where s ranges from 0 at the start to 1 at the end of the path. As s increases from 0 to 1, the robot follows this path.

The motion law, given by the function $s(t)$, complements the path by defining how s changes over time. This relationship determines the robot's position, velocity, acceleration, and jerk along the path.

In summary, a trajectory is defined by combining the geometric path and the temporal motion law, resulting in a time-parameterized sequence of states for the robot.

6.2.1 Jerk

Jerk refers to the rate of change of acceleration; in other words, it measures how quickly an object's acceleration is changing over time. It's the third derivative of position with respect to time, following velocity and acceleration. Its mathematical expression with respect to the motion law is thus $\partial^3 s(t)/\partial t^3$.

In a mechanism, a significant jerk causes vibrations because to create acceleration, an effort must be made, according to the fundamental principle of dynamics. The application of this force and the notion of inertia result in elastic deformation. If the acceleration changes rapidly, then the deformation also changes rapidly; this causes oscillations in the system, and hence vibrations. It's a phenomenon that can be measured, and must be taken into account because of its physical consequences [66].

6.2.2 Jerk's consideration in the design of a law of motion

Let's consider a linear motion, like the one used in Section 4.2: the robot has to bring a solid from a position $\boldsymbol{X}_s$ to a position $\boldsymbol{X}_e$. The equation (4.4) gives the mathematical expression of this motion.

An easy to implement motion law for such trajectory would be a trapezoidal speed law, where there is, for example, a third of the movement in acceleration, a third of the movement at constant speed and a third of the movement in braking.

1. For $t \in \left[0; \frac{T}{3}\right]$.

$$s(t) = \frac{9}{4} x_f \frac{t^2}{T^2} \tag{6.4}$$

2. For $t \in \left[\frac{T}{3}; \frac{2T}{3}\right]$.

$$s(t) = \frac{3}{2} x_f \left[\frac{t}{T} - \frac{1}{6}\right] \tag{6.5}$$

3. For $t \in \left[\frac{2T}{3}; T\right]$.

$$s(t) = \frac{3}{2} x_f \left[-\frac{3}{2}\frac{t^2}{T^2} + 3\frac{t}{T} - \frac{5}{6}\right] \tag{6.6}$$

With such kinematics, the acceleration is a stepped function as shown in Figure 6.5. It is not derivable at transitions, so the jerk is infinite.

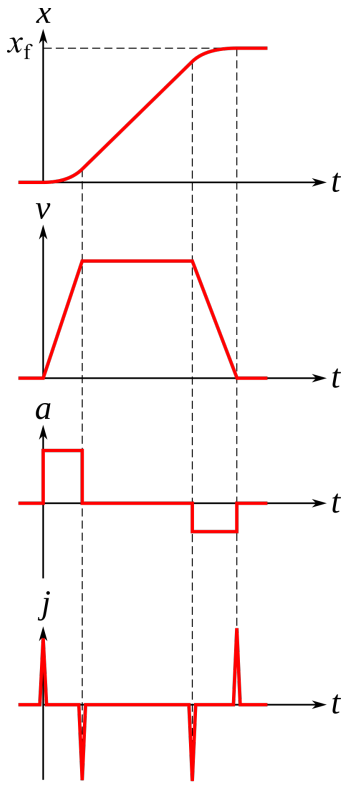


Figure 6.4: Behavior in position, velocity, acceleration and jerk of the trapezoidal speed law (image by Christophe Dang Ngoc Chan[1]).

[1]Christophe Dang Ngoc Chan (Cdang (talk)) — Travail personnel, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=31016928

Laws with theoretical infinite jerk can be used at low speeds, where the vibrations generated are moderate [66]. In the demonstration, one can observe that even a law with infinite jerk (the one used is with a triangular speed profile) is better than none. However, there are law with no infinite jerk. For example, the motion law used for the pick and place trajectory is the modified sinusoidal law or mod-sin (Figure 6.5). It consists of three sinusoidal functions.

1. Sinusoidal function with positive jerk for $t \in \left[0; \frac{T}{8}\right]$.

$$s(t) = \frac{\pi}{\pi + 4} x_f \left[\frac{t}{T} - \frac{1}{4\pi} \sin\left(4\pi \frac{t}{T}\right)\right] \tag{6.7}$$

2. Sinusoidal function with negative jerk for $t \in \left[\frac{T}{8}; \frac{7T}{8}\right]$.

$$s(t) = \frac{\pi}{\pi + 4} x_f \left[\frac{t}{T} - \frac{9}{4\pi} \cos\left(\frac{4\pi}{3}\left(\frac{t}{T} - \frac{1}{8}\right)\right) + \frac{2}{\pi}\right] \tag{6.8}$$

3. Sinusoidal function with positive jerk for $t \in \left[\frac{7T}{8}; T\right]$.

$$s(t) = \frac{\pi}{\pi + 4} x_f \left[\frac{t}{T} - \frac{1}{4\pi} \sin\left(4\pi \frac{t}{T}\right) + \frac{4}{\pi}\right] \tag{6.9}$$

This motion law has a two-part starting phase: a rapid start (0-T/8) followed by a "soft" acceleration phase (T/8-T/2). It then transitions to a "soft" deceleration phase (T/2-7T/8) and ends with a quick braking phase (7T/8-T). This approach is shown in the demonstration because it minimizes required torques while maintaining acceptable jerk values.

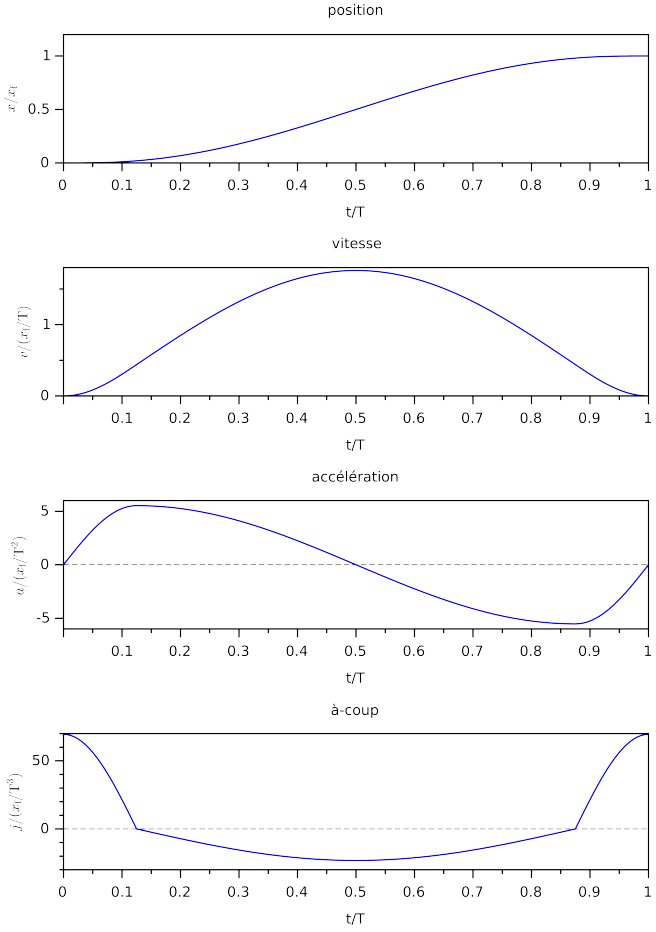


Figure 6.5: Behavior in position, velocity, acceleration and jerk of the modified sinusoidal movement law (image by Christophe Dang Ngoc Chan[2]).

6.2.3 Purpose of the Demonstration

The concept of motion law and its utility can be a bit vague, so this demonstration provides a visual and audible display of different motion laws and their impact on the motor. By the end of the demonstration, students should clearly understand why minimizing jerk preserves the lifetime of a mechanism.

[2]Cdang — Travail personnel, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=72180329

6.3 Improvements

Unfortunately, the trajectory described in equation (6.1) does not allow the pick and place of objects located at different heights, which was a desired feature. In order to remedy to this problem, several solutions were explored, some were close to succeeding, and will be presented hoping to open paths for a future implementation.

One approach involved combining a linear trajectory with the existing pick-and-place trajectory. The resulting trajectory can be expressed as follows:

$$\boldsymbol{X}(t) = \boldsymbol{C} + \boldsymbol{D}\frac{s(t)}{x_f} + A \cdot \boldsymbol{a} \cdot \cos\left(\frac{\pi \cdot s(t)}{x_f}\right) + B \cdot \boldsymbol{b} \cdot \sin\left(\frac{\pi \cdot s(t)}{x_f}\right) \tag{6.10}$$

Where $\boldsymbol{D}$ is the displacement vector. The other parameters do not change from the pick-and-place trajectory. However, this approach did not yield the expected results, likely because the linear trajectory was not perpendicular to the pick-and-place path, causing them to add-up, particularly in the height direction, leading to an undesired position.

Another approach was adapted from [67, 68], using the trajectory equation provided below.

$$\boldsymbol{X}(t) = \boldsymbol{p}_0 + r \begin{bmatrix} A \cdot \cos(\psi) \cdot \cos\left(\frac{\pi \cdot s(t)}{x_f}\right) \\ A \cdot \cos(\psi) \cdot \cos\left(\frac{\pi \cdot s(t)}{x_f}\right) \\ C \cdot \cos\left(\frac{\pi \cdot s(t)}{x_f}\right) + \left[B \cdot \sin\left(\frac{\pi \cdot s(t)}{x_f}\right)\right] / k \end{bmatrix} \tag{6.11}$$

Where the centre of the ellipse is given by the vector $\boldsymbol{p}_0$. r is a linearising function, described below, of which k is a coefficient. ψ represents the angle between the x axis and the vector $\boldsymbol{X}_s + \boldsymbol{X}_e$ projected onto the x-y plane. A, B and C are scaling factors which are calculated from $\boldsymbol{X}_s$, $\boldsymbol{X}_e$ and δ the clearance height. These parameters are illustrated in Figure 6.6.

To enhance clarity in the following equations, let $\boldsymbol{X}_s = \boldsymbol{p}_1$ and $\boldsymbol{X}_e = \boldsymbol{p}_2$.

$$\boldsymbol{p}_0 = \frac{1}{2}(\boldsymbol{p}_1 + \boldsymbol{p}_2) \tag{6.12}$$

$$\psi = \text{atan2}(\boldsymbol{p}_{1y} - \boldsymbol{p}_{2y}, \boldsymbol{p}_{1x} - \boldsymbol{p}_{2x}) \tag{6.13}$$

$$A = \frac{1}{2}\sqrt{(\boldsymbol{p}_{1x} - \boldsymbol{p}_{2x})^2 + (\boldsymbol{p}_{1y} - \boldsymbol{p}_{2y})^2} \tag{6.14}$$

$$B = \delta \tag{6.15}$$

$$C = \frac{1}{2}(\boldsymbol{p}_{1z} - \boldsymbol{p}_{2z}) \tag{6.16}$$

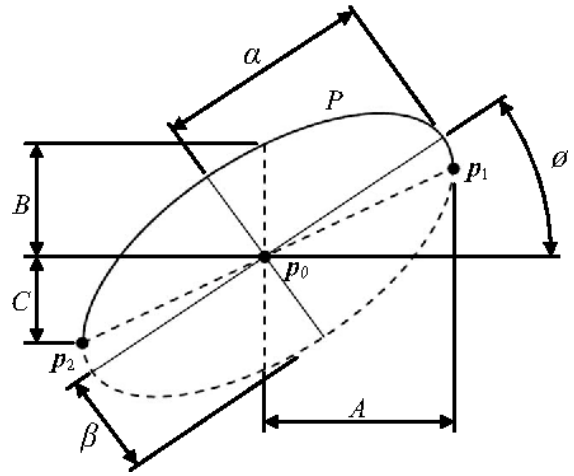


Figure 6.6: Ellipse, showing start and finish points p_1 and p_2, centre point p_0, the scaling factors A, B and C, semi-major axis α, semi- minor axis β, angle to the principal axis ϕ and perimeter P [68].

The parametric equation of an ellipse is non-linear. The displacement along the elliptical path varies non-linearly with the parameter $s(t)$. Before the ellipse can be used effectively as a path primitive for trajectory planning, its parametric equation should first be made approximately linear.

The linearising function r is also a parametric function of $s(t)$ and represents the radius of a scaling ellipse with semi- major axis equal to unity and semi-minor axis k. The amount of scaling applied is governed by k, which in turn is dependent on the eccentricity of the elliptical path ϵ and the constant λ. The effect of λ on the displacement along the elliptical path at equal intervals of $s(t)$ is illustrated in Figure 6.7. A value of $\lambda = 0.14$ was found to give optimum linearisation. α and β correspond to the semi-major and semi-minor axes of the elliptical path respectively and ϕ is the angle between the semi-major axis of the ellipse and the x-y plane [68].

$$r = \frac{k}{\sqrt{1 + (k^2 - 1)\cos^2(\pi s(t))}} \tag{6.17}$$

$$k = 1 - \lambda\epsilon^2 \tag{6.18}$$

$$\epsilon = \sqrt{1 - \frac{\beta^2}{\alpha^2}} \tag{6.19}$$

$$\alpha = \frac{AB}{\sqrt{(C\cos(\phi) - A\sin(\phi))^2 + (B\cos(\phi))^2}} \tag{6.20}$$

$$\beta = \frac{AB}{\sqrt{(C\sin(\phi) + A\cos(\phi))^2 + (B\sin(\phi))^2}} \tag{6.21}$$

$$\phi = \frac{1}{2}\,\mathrm{atan2}(2AC, A^2 - B^2 - C^2) \tag{6.22}$$

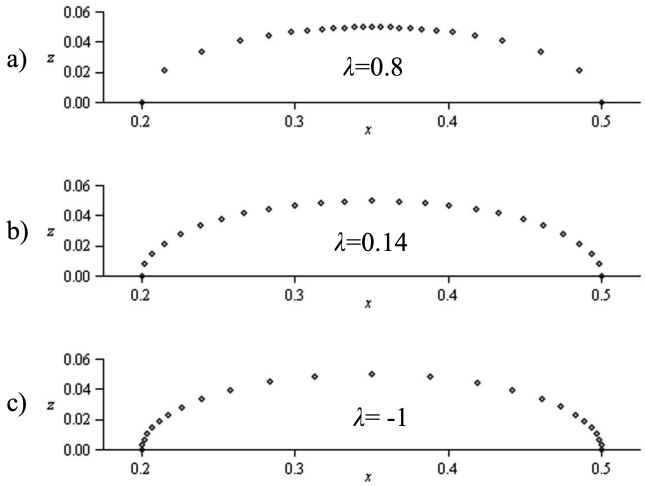


Figure 6.7: Effect of the linearising function r on points distributed at equal intervals of $s(t)$ along the elliptical path. Optimum distribution occurs at $\lambda = 0.14$ [68].

Hopes were high after reviewing those scientific articles, as they closely aligned with the project's needs. However, despite some adaptations, the approach ultimately did not work, possibly due to issues in the C++ implementation. As mentioned earlier, these were the closest attempts to success. Perhaps they will resonate with a future reader, inspiring a solution that can finally resolve this last small remaining challenge with the Delta robot.

Conclusion

This thesis has been a difficult but fascinating project, and the results are something to be proud of. The progress made represents a significant step forward in the development of the Delta robot as an educational tool at UCLouvain. Building on the strong foundation laid by the previous thesis, the Delta robot has been transformed from a prototype into a versatile demonstrator capable of performing a wider range of tasks. The integration of a gripper system into the end effector has expanded the robot's functional capabilities, enhancing its applicability for various educational demonstrations.

The state of the art provided an overview of performance testing practices in the industry, which guided the tests conducted to evaluate key metrics such as position accuracy, repeatability, and flexibility, detailed in Chapters 4 and 5. The implementation and experimental setup were fully explained, confirming that the robot is well-built and effectively controlled — a very encouraging outcome.

Additionally, the development of a graphical user interface (GUI) discussed in Chapter 3, has significantly improved the robot's accessibility, making it easier to operate. This user-friendly interface simplifies the control process, ensuring that the robot can be used efficiently and safely, thereby making it a more versatile tool.

Despite the challenges faced throughout this thesis, perseverance and the support of those around us enabled us to overcome them. The difficulties encountered, more often then not without logical explanations, provided valuable learning experiences as it required innovative problem-solving approaches.

In conclusion, this thesis has successfully advanced the Delta robot project, transforming it into a fully functional pedagogical tool. The anticipation of its future applications is exciting, and it is hoped that the groundwork established here will inspire a young audience to explore the fascinating world of robotics.

Bibliography

[1] R.N.NAGEL N.G.ODREY M.P.GROOVER M.WEISS. *IndustrialRobotics:Technology, Programming and Applications*. Tata McGraw-Hill Edition, 2008.

[2] Khelifa Baizid. "A COMPREHENSIVE STATE-OF-THE-ART ON CONTROL OF INDUSTRIAL ARTICULATED ROBOTS". In: *Journal of the Balkan Tribological Association* 20 (Dec. 2014), 499–521.

[3] Ilias Aghezzaf and Thomas Flamand. *Design of a Delta robot for Pedagogical Purpose*. Ecole polytechnique de Louvain, Université catholique de Louvain: Prom. : Fisette, Paul ; Ronsse, Renaud, 2023. URL: `http://hdl.handle.net/2078.1/thesis:43531`.

[4] Matteo Russo. "Measuring performance: metrics for manipulator design, control, and optimization". In: *Robotics* 12.1 (2022), p. 4.

[5] Michael O Shneier et al. "Measuring and representing the performance of manufacturing assembly robots". In: (2015).

[6] *ANSI/RIA R15.05-2-1992 (R1999): Industrial Robots and Robot Systems - Path-Related and Dynamic Performance Characteristics - Evaluation*. Reaffirmed 1999. 1992. URL: `https://webstore.ansi.org/standards/pcc/ansiriar15051992r1999`.

[7] *ANSI/RIA R15.05-3-1992 (R1999): Industrial Robots and Robot Systems - Reliability Acceptance Testing - Guidelines*. Reaffirmed 1999. 1992. URL: `https://webstore.ansi.org/standards/pcc/ansiriar15051992r1999-1009334`.

[8] *ANSI/RIA R15.05-1-1990 (R1999): Evaluation of Point-to-Point and Static Performance Characteristics of Industrial Robots and Robot Systems*. Reaffirmed 1999. 1990. URL: `https://webstore.ansi.org/standards/pcc/ansiriar15051992r1999-1009334`.

[9] *Manipulating Industrial Robots — Performance Criteria and Related Test Methods*. Revised version from 1998. International Organization for Standardization, 1998.

[10] Adam Jacoff et al. “Comprehensive standard test suites for the performance evaluation of mobile robots”. In: *Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop*. 2010, pp. 161–168.

[11] Cambridge University Press. *Performance*. Accessed: 2024-08-07. 2024. URL: https://dictionary.cambridge.org/nl/woordenboek/engels/performance.

[12] Sarosh Patel and Tarek Sobh. “Manipulator performance measures-a comprehensive literature survey”. In: *Journal of Intelligent & Robotic Systems* 77 (2015), pp. 547–570.

[13] Pyung H Chang. “A Dexterity Measure for the Kinematic Control of Robot Manipulator with Redundany”. In: *Massachusetts Institute of Technology Artificial Intelligence Laboratory* (1988), p. 1.

[14] Bhanoday Reddy Vemula. “Evaluation of Industrial Robot Mechanical Systems for Applications that Require Human-Robot Collaboration”. PhD thesis. Mälardalen University, 2020.

[15] Tsuneo Yoshikawa. “Dynamic manipulability of robot manipulators”. In: *Transactions of the Society of Instrument and Control Engineers* 21.9 (1985), pp. 970–975.

[16] Imed Mansouri and Mohammed Ouali. “The power manipulability–A new homogeneous performance index of robot manipulators”. In: *Robotics and Computer-Integrated Manufacturing* 27.2 (2011), pp. 434–449.

[17] Lisandro J Puglisi et al. “Dimensional synthesis of a spherical parallel manipulator based on the evaluation of global performance indexes”. In: *Robotics and Autonomous Systems* 60.8 (2012), pp. 1037–1045.

[18] S Kucuk and ZAFER Bingul. “Robot workspace optimization based on a novel local and global performance indices”. In: *Proceedings of the IEEE International Symposium on Industrial Electronics, 2005. ISIE 2005.* Vol. 4. IEEE. 2005, pp. 1593–1598.

[19] Serdar Kucuk and Zafer Bingul. “Comparative study of performance indices for fundamental robot manipulators”. In: *Robotics and Autonomous Systems* 54.7 (2006), pp. 567–573.

[20] Junichi Urata et al. “Thermal control of electrical motors for high-power humanoid robots”. In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2008, pp. 2047–2052. DOI: 10.1109/IROS.2008.4651110.

[21] Mohsen Soori, Behrooz Arezoo, and Roza Dastres. “Optimization of energy consumption in industrial robots, a review”. In: *Cognitive Robotics* 3 (2023), pp. 142–157.

[22] Timothy A Zimmerman and Timothy A Zimmerman. *Metrics and key performance indicators for robotic cybersecurity performance analysis*. US Department of Commerce, National Institute of Standards and Technology ..., 2017.

[23] Ahmad Rasdan Ismail et al. "The performance analysis of industrial robot under loaded conditions and various distance". In: *International Journal on Mathematical Models And Methods In Applied Sciences* (2008), pp. 277–284.

[24] Bruno Siciliano, Oussama Khatib, and Torsten Kröger. *Springer handbook of robotics*. Vol. 200. Springer, 2008.

[25] Guilin Yang and I-Ming Chen. "Task-based optimization of modular robot configurations: minimized degree-of-freedom approach". In: *Mechanism and machine theory* 35.4 (2000), pp. 517–540.

[26] Pei Jiang et al. "Energy consumption prediction and optimization of industrial robots based on LSTM". In: *Journal of Manufacturing Systems* 70 (2023), pp. 137–148.

[27] Pranchalee Poonyapak and M John D Hayes. "Towards a predictive model for temperature-induced deformation of an industrial robot". In: *Proc. EuCoMeS, the first European Conference on Mechanism Science, on CD, Obergurgl, Austria*. Vol. 5. 2006.

[28] Fadi Belkhou. *Lecture 5: Robotics*. Accessed: 2024-08-07. 2024. URL: https://ecs-pw-facweb.ecs.csus.edu/~fbelkhou/Lect5Robotics.pdf.

[29] Tian Huang et al. "Optimal design of a 2-DOF pick-and-place parallel robot using dynamic performance indices and angular constraints". In: *Mechanism and machine theory* 70 (2013), pp. 246–253.

[30] Mark W Spong and Mathukumalli Vidyasagar. *Robot dynamics and control*. John Wiley & Sons, 2008.

[31] Robots Done Right. *Robot Repeatability vs. Accuracy*. Accessed: 2024-08-18. n.d. URL: https://robotsdoneright.com/Articles/robot-repeatability-vs-accuracy.html.

[32] Newport Corporation. *Fundamentals of Motion Control*. Accessed: 2024-08-06. n.d. URL: https://www.newport.com/n/motion-control-fundamentals.

[33] Tanio Tanev and Bogdan Stoyanov. "On the performance indexes for robot manipulators". In: *Problems of engineering cybernetics and robotics* 49 (2000), pp. 64–71.

[34] A Harish and G Satish Babu. "Manipulability Index of a Parallel Robot Manipulator". In: *Int. J* 6.6 (2015).

[35] Max Engardt, Axel Heimburger, and Philip Sydhoff. *Manipulability index optimization for a planar robotic arm*. 2012.

[36] Jean-Pierre Merlet. "Jacobian, manipulability, condition number, and accuracy of parallel robots". In: (2006).

[37] Dominique Deblaise, Xavier Hernot, and Patrick Maurine. "A systematic analytical method for PKM stiffness matrix calculation". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE. 2006, pp. 4213–4219.

[38] J Kenneth Salisbury. "Active stiffness control of a manipulator in cartesian coordinates". In: *1980 19th IEEE conference on decision and control including the symposium on adaptive processes*. IEEE. 1980, pp. 95–100.

[39] Renaud Ronsse. *LELME2732 - Robot modelling and control*. Lecture notes from LELME2732, EPL. 2023.

[40] *Printemps des sciences*. July 2024. URL: https://www.sciences.be/printemps-des-sciences/.

[41] Robotran. *Home Robotran - Robotran*. June 2020. URL: https://www.robotran.be/.

[42] Romy A. Letourneau. *Best Practices for Cup Selection*. Accessed: 2024-08-10. 2019. URL: https://blog.robotiq.com/best-practices-for-cup-selection.

[43] Schmalz GmbH. *Theoretical Holding Force of a Suction Cup*. Accessed: 2024-08-09. 2024. URL: https://www.schmalz.com/vacuum-knowledge/the-vacuum-system-and-its-components/system-design-calculation-example/theoretical-holding-force-of-a-suction-cup/.

[44] Şenol Şirin, Enes Aslan, and Gülşah Akincioğlu. "Effects of 3D-printed PLA material with different filling densities on coefficient of friction performance". In: *Rapid Prototyping Journal* 29.1 (2023), pp. 157–165.

[45] Tameson. *Vacuum Suction Cups - Selection Criteria*. Accessed: 2024-08-09. 2024. URL: https://tameson.fr/pages/ventouse-a-vide#selection-criteria.

[46] Electron contributors. *Build cross-platform desktop apps with JavaScript, HTML, and CSS*. https://www.electronjs.org/. Accessed: 2024-08-05. 2023.

[47] Wikipedia contributors. *Electron (software framework) — Wikipedia, The Free Encyclopedia*. Accessed: 2024-08-10. 2024. URL: https://en.wikipedia.org/wiki/Electron_(software_framework).

[48] ShumaTech. *BOSSA (Basic Open Source SAM-BA Application)*. Accessed: 2024-08-10. 2024. URL: https://www.shumatech.com/web/products/bossa.

[49] Inc. Teknic. *UF2 Bootloader - ClearCore Library Documentation*. Accessed: 2024-08-11. 2023. URL: https://teknic-inc.github.io/ClearCore-library/BootloaderMain.html.

[50] Oğuz Düğenci. "Determination of bending and lateral-torsional buckling displacements in castellated and circular opening steel beams by image processing". In: *Structures*. Vol. 56. Elsevier. 2023, p. 104942.

[51] Zhengyou Zhang. "A flexible new technique for camera calibration". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000), pp. 1330–1334.

[52] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc., 2008.

[53] *CALIBRATING A CAMERA: PRINCIPLES AND PROCEDURES*. URL: https://eikosim.com/articles-techniques/etalonnage-camera-principes-et-procedures/.

[54] MathWorks. *Camera Calibration*. Accessed: 2024-08-04. 2024. URL: https://ww2.mathworks.cn/help/vision/ug/camera-calibration.html.

[55] *OpenCV, Camera Calibration*. URL: https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html.

[56] Calib.io. *Understanding Reprojection Errors*. Accessed: 2024-08-04. n.d. URL: https://calib.io/blogs/knowledge-base/understanding-reprojection-errors.

[57] *OpenCV, Detection of ArUco Markers*. URL: https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html.

[58] Sergio Garrido-Jurado et al. "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6 (2014), pp. 2280–2292.

[59] Satoshi Suzuki et al. "Topological structural analysis of digitized binary images by border following". In: *Computer vision, graphics, and image processing* 30.1 (1985), pp. 32–46.

[60] Nobuyuki Ostu. "A threshold selection method from gray-level histograms." In: *IEEE Trans SMC* 9 (1979), p. 62.

[61] Abraham Savitzky and Marcel JE Golay. "Smoothing and differentiation of data by simplified least squares procedures." In: *Analytical chemistry* 36.8 (1964), pp. 1627–1639.

[62] Wikipedia contributors. *Coefficient of determination — Wikipedia, The Free Encyclopedia*. Online; accessed 5-August-2024. 2024. URL: https://en.wikipedia.org/wiki/Coefficient_of_determination.

[63] Long Tian et al. "Full-field bridge deflection monitoring with off-axis digital image correlation". In: *Sensors* 21.15 (2021), p. 5058.

[64] Alain Codourey. *Contribution à la commande des robots rapides et précis: application au robot delta à entraînement direct*. Tech. rep. EPFL, 1991.

[65] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

[66] Contributeurs aux projets Wikimedia. *À-coup*. May 2024. URL: https://fr.wikipedia.org/wiki/%C3%80-coup.

[67] Myong Song Choe et al. "An Approach for Elliptical Trajectory Planning with Vertical Straight Line Segments of Pick-and-Place Robot Operation with Height Clearance". In: *Mathematical Problems in Engineering* 2023.1 (2023), p. 7419178.

[68] René J. Moreno Masey et al. "Elliptical point to point trajectory planning using electronic cam motion profiles for high speed industrial pick and place robots". In: *2009 IEEE Conference on Emerging Technologies & Factory Automation* (2009), pp. 1–8. URL: https://api.semanticscholar.org/CorpusID:16313497.

Appendix A

Specifications

	Delta Robot Specifications	Date : 07/04/2023 Version : 6

Context: For the TFE23-249 we need to develop a 3 d.o.f Delta robot. The robot will be used for pedagogical purpose by the teaching staff.

Date	**Main Functions**
7/10/2022	MF1 : Accuracy, Repeatability and Resolution of the robot
7/10/2022	MF2 : Max payload
7/10/2022	MF3 : Speed and Acceleration
7/10/2022	MF4 : Perform a range of tasks
10/10/2022	MF5 : Replaceability
	Optional
10/10	MF6 : Integrate computer vision
20/10	MF7 : 4th degree of freedom
	Functional Requirement (FR) criteria and levels
7/10/2022	FR 1.1 : Accuracy: the robot should be able to reach a target within an error margin of 1 mm
7/10/2022	FR 1.2 : Repeatability: the distance in the 0xyz space between different consecutive measurements of a reached target should be in the interval of ± 0.5mm
2/3/2023	FR 1.3 : Resolution: the minimal distance the robot should be able to travel is 0.6mm
7/10/2022	FR 2.1 : The robot should be able to carry a static payload of 1 kg
2/3/2023	FR 2.2 : The robot should be able to carry a dynamic payload of 250 g
7/10/2022	FR 3.1 : The platform should be able to reach a maximum speed of 3 m/s
7/10/2022	FR 3.2 : The platform should be able to reach a maximum acceleration of 10 g's
7/10/2022	FR 4.1 : Pick and place task to show the acceleration, speed, accuracy and repeatability of the robot
11/10/2022	FR 4.2 : Path following task to show the acceleration and speed of the robot
7/10/2022	FR 4.3 : Drawing task to show the accuracy and precision of the robot
7/10/2022	FR 4.4 : Target reaching task in order to show the accuracy and repeatability of the robot
20/10/2022	FR 4.5 : Race between the user and the robot to show the speed and acceleration of the robot
10/10/2022	FR 5.1 : If any mechanical or electrical failure occurs, the exchange of pieces should be easy to do
10/10/2022	FR 6.1 : The robot should be able to recognize objects and act accordingly
20/10/2022	FR 7.1 : Accuracy: the robot should be able to rotate an object within an error margin of 4°
20/10/2022	FR 7.2 : Repeatability: different measurements of the same rotation of the end effector along the z-axis should be in the interval of ± 2°
20/10/2022	FR 7.3 : Resolution: the minimal angle the end effector should be able to rotate is 2°
	Main constraints
7/10/2022	MC1 : Price of conception
7/10/2022	MC2 : Workspace dimensions
7/10/2022	MC3 : Power consumption
7/10/2022	MC4 : Working temperature range
7/10/2022	MC5 : Noise
7/10/2022	MC6 : Safety
18/10/2022	MC7 : Usability
20/20/2022	MC8 : Transportability

7/10/2022	MC9: Future master theses
	Constraint Requirement (CR) criteria and levels
7/4/2023	CR 1.1 : The robot's conception price should be less than 4000 €
7/10/2022	CR 2.1 : The robot's workspace dimensions for the 0xy axis should be 300 to 300 mm
2/3/2023	CR 2.2 : The robot's workspace dimension for the 0z axis should be 150 mm
10/10/2022	CR 2.3 : There should be no singularities in the workspace
7/10/2022	CR 3.1 : The maximum electrical power consumption of the robot should be 2000 W
7/10/2022	CR 4.1 : The robot should be able to work in a range of temperatures of 0 to 45 °C
7/10/2022	CR 5.1 : The robot's noise should be less than 60 dB
7/10/2022	CR 6.1 : The robot should pass the risk analysis of UCLouvain
20/10/2022	CR 7.1 : Usable without any risk of overheating for normal usage
11/10/2022	CR 7.2 : Usable without any terminating mechanical failure for 10 years
7/10/2022	CR 7.3 : Easy to use, the activation of the robot should not require complex actions
7/10/2022	CR 7.4 : Allow clean view of the robot for filming and viewing during sessions
20/10/2022	CR 7.5 : Have a clear documentation for the usage of the robot
20/10/2022	CR 8.1 : The robot should be able to be transported in a utility vehicle
20/10/2022	CR 8.2 : The robot and its frame should be able to pass through doorways 75cm wide
7/10/2022	CR 9.1 : Allow further work for future master theses by considering the potential upgrades on the robot
7/10/2022	CR 9.2 : Have a clear documentation for future work on the robot

Appendices :

Specifications	**Value**	**Unit**
FR1.1 : Accuracy	1	mm
FR1.2 : Repeatability	± 0.5	mm
FR1.4 : Resolution	0.6	mm
FR2.1 : Max static load	1	Kg
FR2.1 : Max dynamic load	250	g
FR3.1 : Max speed	3	m/s
FR3.2 : Max acceleration	10	g's
FR7.1 : Accuracy	π/45	rad
FR7.2 : Repeatability	± π/90	rad
FR7.3 : Resolution	π/90	rad
CR1.1 : max Price	4000	€
CR2.1 : Workspace 0xy plane	300 x 300	mm
CR2.2 : Workspace 0z axis	150	mm
CR3.1 : Max power consumption	2000	W
CR4.1 : Temperature range	0 – 45	°C
CR5.1 : Noise	60	dB
CR7.2 : Warranty	10	years
CR8.2 : Max width	750	mm

Appendix B

Detailed budget

Pièce	Référence	Provenance	Prix HTVA	Nombre	Coûts HTVA
Ventouse	OGVM-30-S-N-G14F	Festo	18.24€	1	18.24€
Raccord enfichable	QS-1/4-4	Festo	10.9€	1	10.9€
Venturi	VN-05-M-I3-PQ2-VQ2	Festo	22.53€	1	22.53€
Capteur de pression	SDE5-V1-FP-Q6-P-M8	Festo	75.24€	1	75.24€
Câble de liaison	NEBA-M8G3-U-5-N-LE3	Festo	6.0€	1	6.0€
Électrodistributeur	MHJ10-S-2,5-QS-6-HF/LP	Festo	76.96€	1	76.96€
					Coût total HTVA : 209,87€

Appendix C

Table for test results

Table C.1: Pose accuracy and Pose repeatability

Load	Velocity	P_1		P_2		P_3		P_4		P_5	
100%	100%,	$AP_P = 0.17$	$RP = 0.3$	$AP_P = 0.12$	$RP = 0.18$	$AP_P = 0.11$	$RP = 0.36$	$AP_P = 0.14$	$RP = 0.22$	$AP_P = 0.096$	$RP = 0.19$
	50%	$AP_P = 0.073$	$RP = 0.2$	$AP_P = 0.11$	$RP = 0.16$	$AP_P = 0.15$	$RP = 0.54$	$AP_P = 0.094$	$RP = 0.18$	$AP_P = 0.084$	$RP = 0.17$

All poses in the table above are expressed in millimeters.

Appendix D

Laser

Illustration may differ

Ordering information

Type	Part no.
OD2-P250W150U0	6036646

Other models and accessories → www.sick.com/OD_Value

Detailed technical data

Mechanics/electronics

Supply voltage V_S	DC 18 V ... 24 V
Power consumption	≤ 2.88 W [1]
Warm-up time	≤ 30 min
Housing material	Plastic (PBT)
Window material	Plastic (PMMA)
Connection type	Male connector, M12, 8-pin, swivel connector unit
Indication	Distance bar graph, up to 8 status LEDs
Weight	70 g
Dimensions (W x H x D)	20.4 mm x 60 mm x 50 mm
Enclosure rating	IP67
Protection class	III

[1] Without load, with current output.

Safety-related parameters

$MTTF_D$	101 years
DC_{avg}	0%

Performance

Measuring range	100 mm ... 400 mm [1]
Target	Natural objects

[1] 6% ... 90% remission factor.
[2] Measurement on 90 % remission (ceramic, white).
[3] At averaging function medium.
[4] Constant ambient conditions.
[5] When calibrated in the application regularly.
[6] Wavelength: 655 nm, max. output: 1 mW.

Repeatability	75 µm [2) 3) 4)]
Linearity	± 750 µm [2) 3) 5)]
Response time	≥ 1 ms
Measuring frequency	≤ 1.33 kHz [1)]
Output time	≥ 0.75 ms
Light source	Laser, red visible red light
Laser class	2 (IEC 60825-1:2014, EN 60825-1:2014) [6)]
Typ. light spot size (distance)	1.8 mm x 3.5 mm (250 mm)
Additional function	Mean-value setting 1 ... 64x, automatic sensitivity adjustment, Analog outputs can be taught in, Invertable analog output, Teach-in of digital output, Invertable switching output, multifunctional input: laser-off / external teach-in / trigger, switching mode: distance to object (DtO), switching mode: window (Wnd)

[1)] 6% ... 90% remission factor.
[2)] Measurement on 90 % remission (ceramic, white).
[3)] At averaging function medium.
[4)] Constant ambient conditions.
[5)] When calibrated in the application regularly.
[6)] Wavelength: 655 nm, max. output: 1 mW.

Interfaces

Digital output	
Number	2 [1)]
Type	PNP
Maximum output current I_A	≤ 100 mA
Analog output	
Number	1
Type	Voltage output
Voltage	0 V ... 10 V, > 10,000 Ω
Resolution	16 bit
Multifunctional input (MF)	1 x MF [2)]

[1)] PNP: HIGH = V_S - (< 2 V) / LOW = < 2 V; NPN: HIGH = < 2 V / LOW = V_S.
[2)] MF can be used as laser-off, trigger, external teach-in, or deactivated; response time ≤ 3 ms.

Ambient data

Ambient temperature, operation	–10 °C ... +40 °C
Ambient temperature, storage	–20 °C ... +60 °C
Relative air humidity (non-condensing)	35 % ... 95 %
Temperature drift	± 0.08 % FS/K (FS = Full Scale = Measuring range of sensor)
Typ. Ambient light immunity	Artificial light: ≤ 3,000 lx Sunlight: ≤ 10,000 lx
Vibration resistance	10 Hz ... 55 Hz (amplitude 1.5 mm, x-, y-, z-axis 2 hours each)
Shock resistance	50 G (x, y, z axis 3 times each)

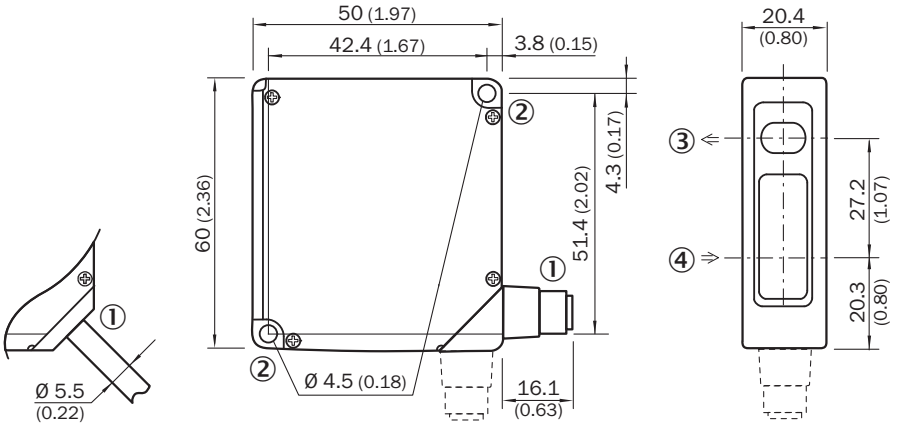
Classifications

eCl@ss 5.0	27270801
eCl@ss 5.1.4	27270801

eCl@ss 6.0	27270801
eCl@ss 6.2	27270801
eCl@ss 7.0	27270801
eCl@ss 8.0	27270801
eCl@ss 8.1	27270801
eCl@ss 9.0	27270801
eCl@ss 10.0	27270801
eCl@ss 11.0	27270801
eCl@ss 12.0	27270916
ETIM 5.0	EC001825
ETIM 6.0	EC001825
ETIM 7.0	EC001825
ETIM 8.0	EC001825
UNSPSC 16.0901	41111613

Dimensional drawing (Dimensions in mm (inch))

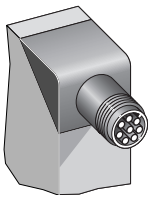
OD2-x250W150xx



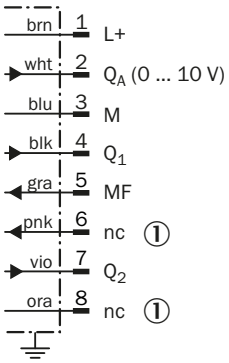
① 2 m cable or M12 connector; 90° rotatable
② Mounting hole, Ø 4.5 mm
③ Optical axis, sender
④ Optical axis, receiver

Connection type

OD2-xxxxxxA0 OD2-xxxxxxC0 OD2-xxxxxxI0 OD2-xxxxxxU0 connector M12, 8-pin



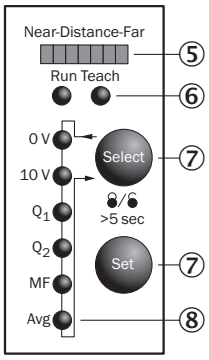
Connection diagram



① Not assigned

Adjustment possible

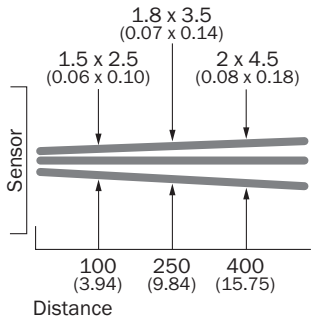
OD2-xxxxxxUx



⑤ Distance indicator
⑥ Mode indicator (Run/Teach)
⑦ Control elements
⑧ Status indicator in- and outputs (Run-mode)/menu indicator (Teach-mode)

Light spot size

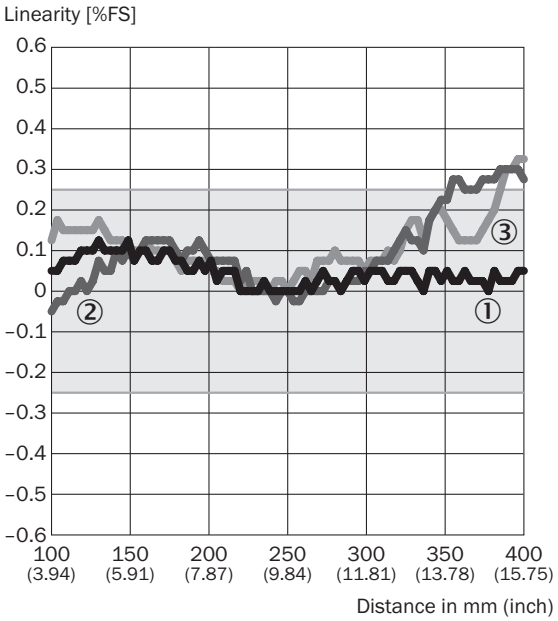
OD2-x250W150xx



All dimensions in mm (inch)

Linearity

OD2-x250xxxxx



① White ceramic
② Black paper
③ Stainless steel

Recommended accessories

Other models and accessories ➜ www.sick.com/OD_Value

	Brief description	Type	Part no.
Mounting brackets and plates			
	Stainless-steel mounting bracket, stainless steel	BEF-WN-OD1000	4089813
Plug connectors and cables			
	Head A: female connector, M12, 8-pin, straight Head B: Flying leads Cable: Sensor/actuator cable, special color code, PVC, shielded, 2 m	DOL-1208-G02MF	6020663

Appendix E

High Speed Camera

PRODUCT DATASHEET

Mini UX

FASTCAM series by Photron

FASTCAM Mini UX50 / UX100

1.3-Megapixel CMOS Sensor:
Mini UX50
1280 x 1024 pixels at 2,000fps
1280 x 800 pixels at 2,500fps
Mini UX100
1280 x 1024 pixels at 4,000fps
1280 x 1000 pixels at 5,000fps
1280 x 800 pixles at 6,250fps

Maximum Frame Rate:
160,000fps (Mini UX50 type 160K)
204,800fps (Mini UX100 type 200K)
800,000fps (Mini UX100 type 800K)

Class Leading Light Sensitivity:
ISO 10,000 monochrome
ISO 5,000 color

Global Electronic Shutter:
Minimum Shutter speed 3.9µs (to 1µs dependent on frame rate selection)

Dynamic Range (ADC):
12-bit monochrome, 36-bit color

Compact and Lightweight:
120mm (H) x 120mm (W) x 93mm (D)
4.72″ (H) x 4.72″ (W) x 3.66″ (D)
Weight: 1.5Kg (3.13 lbs.)

Internal Recording Memory:
8GB, 16GB, or 32GB

Fast Gigabit Ethernet Interface:
Provides high-speed image download to a standard notebook/PC

Flexible Frame Synchronization:
Frame rate may be synchronized to external unstable frequencies

1-Inch C-mount Compatible Sensor Size:
Also supplied with integrated Nikon G-type lens mounts.

High-G Rated:
Suitable for application in high-G environments; operation tested to 100G, 10ms, 6-axes

COMPACT HIGH-SPEED CAMERA SYSTEM

For use with a wide range of general scientific and industrial applications the Photron FASTCAM Mini UX high-speed camera provides outstanding imaging performance at a very attractive price performance ratio.

Two Mini UX camera models provide 1.3-megapixel (1280 x 1024 pixels) image resolution with frame rates up to 2,000fps from the Mini UX50 and 4,000fps from the Mini UX100. Both models are available with recording memory options up to 32GB providing extended recording times and triggering flexibility.

Using innovative proprietary CMOS image sensor technology, the FASTCAM Mini UX achieves high light sensitivity from a small image sensor (10µm pixel pitch) through the utilization of microlenses to increase effective Fill Factor. At maximum image resolution the image sensor is fully compatible with readily available 1-inch C-mount lenses offering a wide choice of small, light weight, rugged and high aperture objective lenses.

The FASTCAM Mini UX features a rugged design suitable for operation in high shock and vibration environments and a compact camera body (120mm x 120mm x 93mm) weighing just 1.5kg.This small and rugged camera design makes the FASTCAM Mini UX ideally suited to on-board and off-board automotive safety testing and many other applications where a compact size and compatibility with standard optical systems is required.

Standard operational features of the FASTCAM Mini UX include a Gigabit Ethernet Interface for reliable system control with high-speed data transfer to PC, and the ability to remotely switch off cooling fans to eliminate vibrations when recording at high magnifications.

Image Sensor Technical Data

Light Sensitivity:

FASTCAM MINI UX	
Monochrome models	ISO 10,000
Color models	ISO 5,000

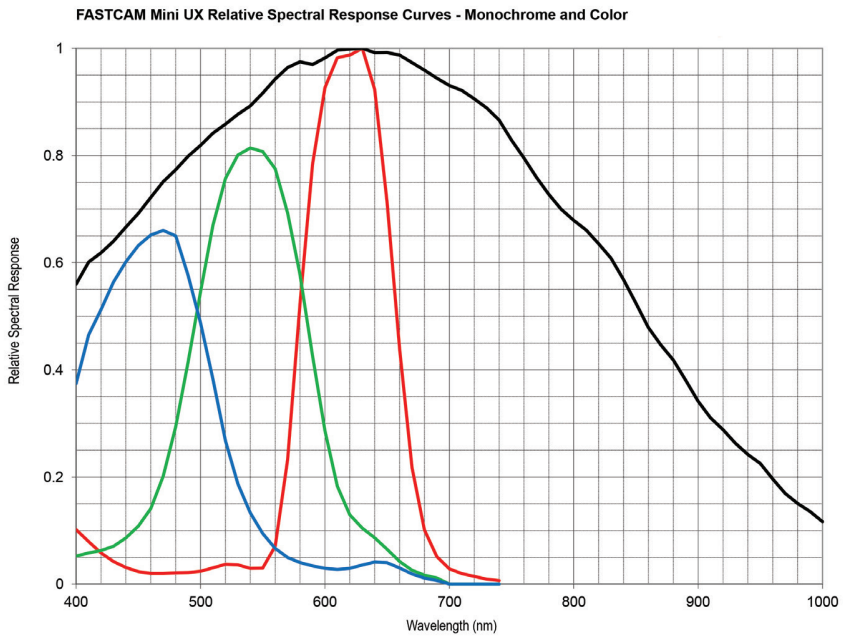
Monochrome sensors used in the FASTCAM Mini UX cameras are supplied without an IR absorbing filter, extending the camera spectral response beyond 900nm. When the sensitivity of the FASTCAM Mini UX camera is measured to tungsten light including near IR response an equivalent value of ISO 25,000 is obtained.

Image Sensor:
The FASTCAM Mini UX system uses an advanced CMOS image sensor optimized for light sensitivity and high image quality that is unique to Photron.

A 10-micron pixel pitch gives a sensor size at full image resolution of 12.8 x 10.24mm (diagonal 16.39mm).

Lenses designed for both 1-inch C-mount and FX / DX (APS-C digital SLR) formats are compatible with the FASTCAM Mini UX at full image resolution.

Sensor Type	Proprietary Design Advanced CMOS
Maximum Reolution (pixels)	1280 x 1024 pixels
Sensor Size / Diagonal	12.80 (H) x 10.24mm (V) / 16.39mm (D)
Pixel Size (microns)	10µm x 10µm
Quantum Efficiency	62.6% at 630nm
Fill Factor	80%
Color Matrix	Bayer CFA (single sensor)
Light Sensitivity	ISO 10,000 monochrome ISO 5,000 color (monochrome sensor equivalent ISO 25,000 including near IR response)
Shutter	Global Electronic Shutter 3.9µs up to 1.01µs at maximum frame rates



Camera Performance Specifications

Model	Mini UX50	Mini UX100
Full Frame Performance	2,000fps 1280 x 1024 pixels	4,000fps 1280 x 1024 pixels
Maximum Frame Rate	Type 160K: 160,000fps (1280 x 8 pixels)	Type 200K: 200,000fps (640 x 8 pixels) Type 800K: 800,000fps (640 x 8 pixels) *
Minimum Exposure	Global electronic shutter 3.9µs (1.01µs at maximum frame rates)	
Dynamic Range (ADC)	12-bit monochrome 36-bit color	
Memory Capacity Options	8GB: 4,365 frames at full resolution 16GB: 8,734 frames at full resolution 32GB: 17,472 frames at full resolution	
Memory Partitions	Up to 64 memory segments	
Region of Interest	Selectable in steps of 128 pixels (horizontal) x 8 pixels (vertical) - minimum 640 x 8	
Trigger Inputs	Selectable +/- TTL 5V and switch closure	
Trigger Delay	Programmable on selected input / output triggers: 100ns resolution	
Input / Output	Input: Trigger (TTL/Switch), sync, ready, event, IRIG Output: trigger, sync, ready, rec, exposure	
Trigger Modes	Start, end, center, manual, random	
Time Code Input	IRIG-B	
External Sync	+/- TTL 5Vp-p Variable frequency sync	
Camera Control Interface	High-speed Gigabit Ethernet	
Image Data Display	Frame rate, shutter speed, trigger mode, date/time, status, real time / IRIG time, frame count, resolution	
Saved Image Formats	BMP, TIFF, JPEG, PNG, RAW, RAWW, MRAW, AVI, WMV, FTIF, MOV - Images can be saved with or without image data and in 8-bit, 16-bit or 36-bit depth of sensor where supported	
Supported OS	Microsoft Windows operating system including: XP, Vista, 7, 8, 8.1, 10, 11 (32/64-bit)	

*** Frame rates above 225,000fps may be subject to export control regulations in some areas**

High-Speed Gigabit Ethernet Interface:
The FASTCAM Mini UX camera system is equipped with a high-speed Gigabit Ethernet Interface to provide reliable network communication and fast download of image data.

Dedicated I/O:
A dedicated BNC connection for a contact closure hardware trigger input is provided. In addition, two programmable inputs and two programmable output channels provide direct connection for common tasks such as synchronization of multiple cameras and operation in conjunction with Data Acquisition (DAQ) hardware.

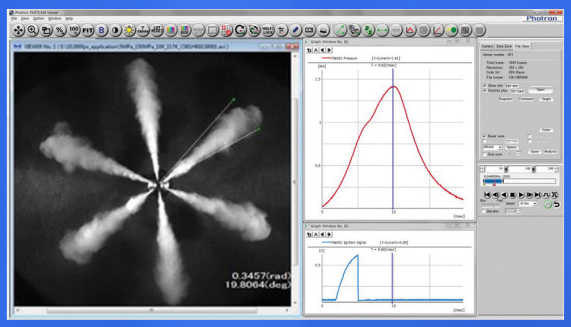
Nikon G-Type Compatible Lens Fitting:
The FASTCAM Mini UX camera is equipped with an objective lens mount compatible with readily available Nikon G-type lenses. Controls provided within the lens mount allow the control of lens aperture on lenses without external iris control.

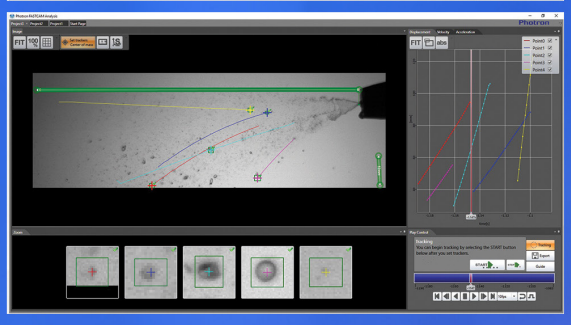
Camera Operation Features

Frame Synchronization	Accurate frame synchronization with other cameras and with external and unstable frequencies.
Memory Partitions	Up to 64 memory segments allow multiple events to be stored in camera memory before downloading, with automatic progression to the next available partition.
Low Light Mode	Operation at minimum frame rate with separately adjustable shutter time to allow easy camera set-up and focus in ambient lighting.
IRIG Phase Lock	Enables multiple cameras to be synchronized together with other instrumentation equipment or to a master external time source.
Internal Time Delay Generator	Allows programmable delays to be set on input and output triggers; 100ns resolution.
Event Markers	Up to ten user-entered event markers to define specific events within the recorded image sequence .
Automatic Download	The system can be set to automatically download image data to the control PC and, when download is complete to re-arm in readiness for the next trigger with automatically incremented file names.
Software Binning	Virtual pixel binning (2x2, 4x4 etc.) allows increased light sensitivity with reduced image resolution without changing camera field of view.

Operation Software Features

Image Calibration	2D image calibration allows the measurement of distance and angle from the image. A calibration grid overlay can be superimposed on the image.
Image Overlay	A stored reference image may be overlaid on the live image to allow accurate camera positioning to achieve the same view as a previous test.
Import of Multiple Image Sequences	Multiple image sequences can be loaded and simultaneously replayed. Timing of image sequences can be adjusted to create a common time reference. Time based synchronization allows images captured at different frame rates to be synchronized.
High Dynamic Range Mode	Making use of the full sensor dynamic range, HDR mode allows enhanced detail in both light and dark areas of an image to be displayed simultaneously.
Motion Detector	In order to highlight subtle changes in an image, Motion Detector allows a reference image to be subtracted from a recorded sequence. Details including propagation of shock waves and surface changes during impact can be visualized using the feature.
Line Profile	A line profile representing grey levels along a line drawn across any region of the image is displayed. In live mode the Line Profile can be used to ensure optimum image focus is achieved.
Histogram	A histogram displaying grey levels within a user-defined image area is displayed. In live mode the Histogram can be used to ensure that optimum exposure levels are set for the scene being recorded.

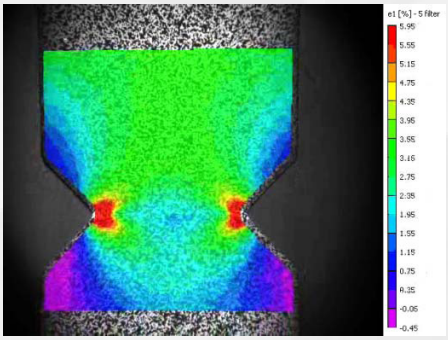




Photron FASTCAM Viewer:
Photron FASTCAM Viewer software (PFV) has been designed to provide an intuitive and feature rich user interface for the control of Photron high-speed cameras, data saving, image replay and simple motion analysis. Advanced operation menus provide access to features for advanced camera operation and image enhancement. Tools are provided to allow image calibration and easy measurement of angles and distances from image data. Also included are a C++ SDK and wrappers for LabView and MATLAB ®.

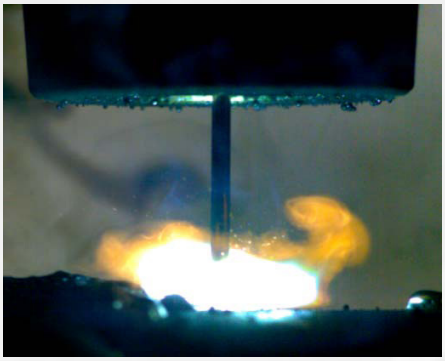
An optional software plug-in module provides synchronisation between Photron high- speed cameras and data acquired through National Instruments data acquisition systems. Synchronised data captured by the DAQ system provides waveform information which can be viewed alongside high-speed camera images.

Photron FASTCAM Analysis:
PFV software allows image sequences to be exported directly to optional Photron FASTCAM Analysis (PFA) Motion Analysis software. This entry level Motion Analysis software with an on screen ‘step by step guide’ function launches automatically from Photron FASTCAM Viewer software, and provides automated tracking of up to 5 points using feature or correlation tracking algorithms for the automated analysis of motion within an image sequence.

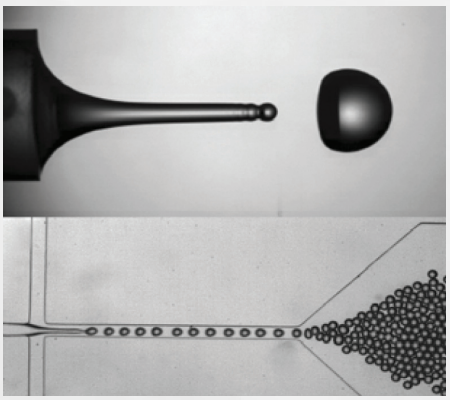


Material Testing & DIC

Biomechanics



Welding & Plasma Research



Fluid Dynamics & Microfludics

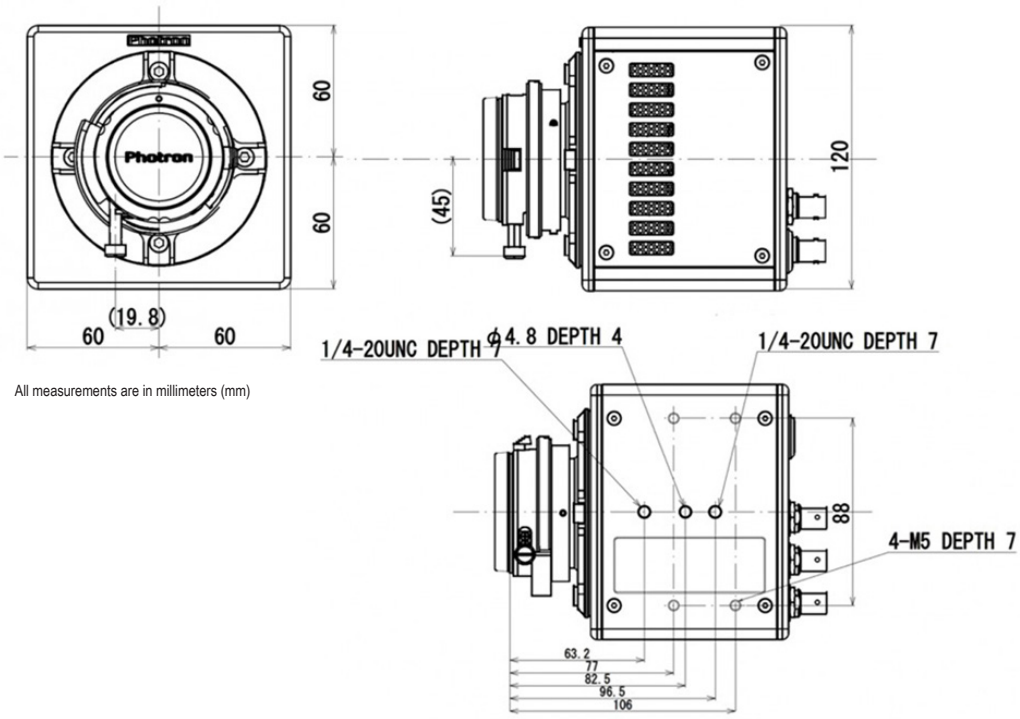
Mini UX100							
Resolution	**Frame Rate**	**8GB**		**16GB**		**32GB**	
(h x v pixels)	Max fps	Frames	Time (sec)**	Frames	Time (sec)**	Frames	Time (sec)**
1280 x 1024	4,000	4,365	1.09	8,734	2.18	17,472	4.37
1280 x 800	6,250	5,587	0.89	11,179	1.79	22,364	3.58
1280 x 720	6,400	6,208	0.97	12,421	1.94	24,849	3.88
1280 x 512	8,192	8,734	1.07	17,468	2.13	34,944	4.27
1280 x 120	40,960	37,248	0.91	74,531	1.82	149,096	3.64
1024 x 1024	4,000	5,458	1.36	10,917	2.73	21,840	5.46
896 x 896	5,120	7,129	1.39	14,259	2.78	28,526	5.57
1280 x 24	204,800	186,242	0.91	372,656	1.82	745,483	3.64

Mini UX50							
Resolution	**Frame Rate**	**8GB**		**16GB**		**32GB**	
(h x v pixels)	Max fps	Frames	Time (sec)**	Frames	Time (sec)**	Frames	Time (sec)**
1280 x 1024	2,000	4,365	2.18	8,734	4.37	17,472	8.74
1280 x 800	2,500	5,587	2.23	11,179	4.47	22,364	8.95
1280 x 720	3,200	6,208	1.94	12,421	3.88	24,849	7.77
1280 x 512	4,000	8,734	2.18	17,468	4.37	34,944	8.74
1280 x 120	20,480	37,248	1.82	74,531	3.64	149,096	7.28
1024 x 1024	2,000	5,458	2.73	10,917	5.46	21,840	10.92
896 x 896	2,500	7,129	2.85	14,259	5.70	28,526	11.41
1280 x 24	102,400	186,242	1.82	372,656	3.64	745,483	7.28

* Specifications subject to change without notice.

** Recording time is an estimate and may be different depending on recording conditions and settings.

Mechanical & Environmental Specifications



All measurements are in millimeters (mm)

Compatibility with Specialist Lens Systems:
A combination of small physical size, low weight and high light sensitivity allows the FASTCAM Mini UX to be coupled to a range of optical systems such as scientific and long distance microscopes, rigid endoscopes or borescopes and image intensifiers for applications ranging from imaging flows in microfluidic devices to combustion diagnostics.

Rugged and Compact Design:
The FASTCAM Mini UX is engineered for use in environments where it may be subject to extreme mechanical shock and vibration. The system has been fully tested for repeated operation at 100G, 10ms, 6-axes.

Small Physical Size:
The small physical size and weight of the Mini camera range allows the use of conventional opto-mechanical hardware for rigid and stable mounting of multiple cameras, and for the location of cameras in space limited locations.

Specifications subject to change without notice.

Mechanical and Environmental Specifications	
Mechanical	
Lens Mount	F-mount (G-type lens compatible) and C-mount provided
Camera Mountings	4 x 1/4 - 20 UNC (base and top), 4 x M5 (base)
External Dimensions	
Camera Body (excluding protrusions)	120mm (H) x 120mm (W) x 93mm (D) 4.72" (H) x 4.72" (W) x 3.66" (D)
Weight	
Camera Body	1.5kg (3.31lbs)
Environmental	
Operating Temperature	0 to 40C, 32˚ to 104˚F
Storage Temperature	-20 to 60C, -4˚ to 140˚F
Humidity	85% or less (non condensing)
Cooling	Internal fan cooling (fan-off mode supported)
Operational Shock	100G, 10ms, 6-axes
Power	
AC Power (with supplied adapter)	100 to 240V, 50 to 60Hz
DC Power	22 to 32V, 40VA

PHOTRON USA, INC.
9520 Padgett Street, Suite 110
San Diego, CA 92126
USA

Tel: 858.684.3555 or 800.585.2129
Fax: 858.684.3558
Email: image@photron.com
www.photron.com

PHOTRON EUROPE LIMITED
The Barn, Bottom Road
West Wycombe
Bucks. HP14 4BS
United Kingdom

Tel: +44 (0) 1494 481011
Fax: +44 (0) 1494 487011
Email: image@photron.com
www.photron.com

PHOTRON (Shanghai)
Room 20C, Zhao-Feng
World Trade Building
No. 369, JiangSu Road
Chang Ning District
Shanghai, 200050 China
Tel: +86 (21) 5268-3700
Fax: +86 (21) 5268-3702
Email: info@photron.cn.com
www.photron.cn.com

PHOTRON LIMITED
21F, Jinbocho Mitsui Bldg.
1-105 Kanda Jimbocho
Chiyoda-ku, Tokyo 101-0051
Japan

Tel: +81 (3) 3518-6271
Fax: +81 (3) 3 3518-6279
Email: image@photron.co.jp
www.photron.co.jp

REV# 23.02.17

Appendix F

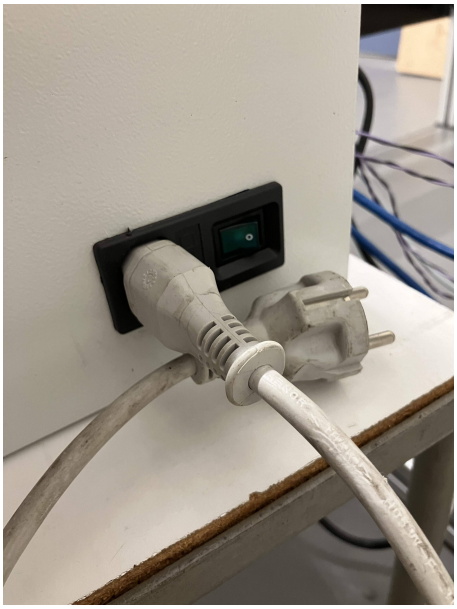
How to use the Delta robot

This appendix serves as a guide for anyone interested in testing the Delta robot, whether they have no prior knowledge and wish to use the interface or want to implement their own trajectories. It also aims to list common errors that may occur while operating the robot and provide solutions for each.

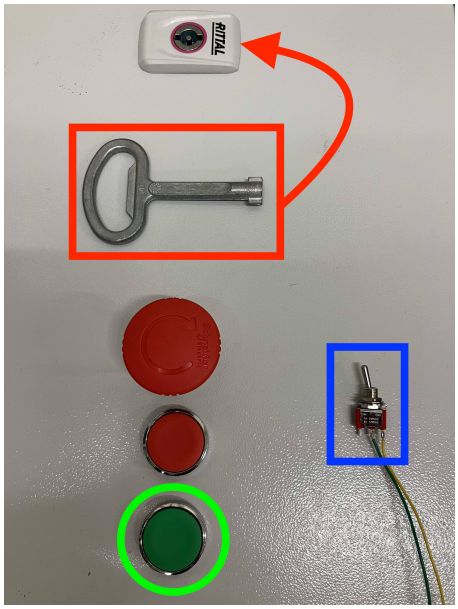
F.1 Preparation Steps

This section outlines the initial steps required to work with the Delta robot. These steps should be completed before proceeding with any of the following procedures.

1. Plug the power cord of the box connected to the Delta robot into a socket (Figure F.1a).
2. Turn on the box by switching the switch on its side (Figure F.1a).
3. Power on the motors by pressing the green button on the box (Figure F.1b).
4. Connect the lab's computer to the Clearcore board using the USB-B cable.



(a) The power supply of the box, showing the plug for power and the switch to turn it on



(b) Highlighted in green is the button to start the motors, in blue is the switch to stop any process in progress, and in red is the key to open the box and access the Clearcore board

Figure F.1: Key components to be familiar with when working with the Delta robot

F.2 Launching Pre-defined Trajectories with the GUI

To launch the pre-defined tasks or trajectories we have implemented, follow these steps using the interface:

1. Launch the "Delta GUI" application (Figure F.2) on the lab's computer.

Figure F.2: Delta robot's GUI logo

2. If the connection to the Clearcore board via the USB-B cable is successful, the screen should display the message: **Status: Device Connected**.
3. You can now select and launch the desired sequence by clicking on its name in the interface.

F.3 Programming the Clearcore board using ATMEL studio

If you want to implement your own trajectories for the Delta robot, you can program the Clearcore board using ATMEL Studio (now known as Microchip Studio). Useful links:

- TEKNIC gives a manual to install ATMEL studio in `https://teknic.com/downloads/`
- The whole library of the Clearcore board can be found following this link: `https://teknic-inc.github.io/ClearCore-library/index.html`
- The Github of the Clearcore board can be found here: `https://github.com/Teknic-Inc/ClearCore-library`

TEKNIC offers numerous examples to help users create their own code. These examples are well-commented and can be useful for understanding how to use various modes. Follow these steps to create and run a project:

1. On the home page of ATMEL Studio, click on "File", then "Open", and select "Project or Solution".
2. Double-click the .atsln file of the project you want to run.
3. When prompted with three windows asking if you want to regenerate pack information, click "No" each time. Clicking "Yes" will replace the .xml file, which may prevent your code from running.
4. Right-click the .cpp file you want to run and select "Set as Startup Project".
5. Select the .cpp file you want to modify.
6. After making modifications, click "Start Without Debugging" or press Ctrl+Alt+F5 to run your code.
7. If your PC is connected to the Clearcore board via a USB-B cable, the code will be uploaded to the board, and the motors will start moving.

F.4 Common Errors

1. **The robot starts moving when the motors are turned on.**
   If the movement is unintended, use the designated switch to stop the process and return the robot to its resting position (Figure F.1b).

2. **The robot is stuck in the homing position.**
   If the robot is not moving from the homing position and the switch to end the sequence does not work, the code on the board might be faulty, causing the card to freeze. To resolve this, quickly double-click the reset button on the Clearcore board. This will enter Bootloader mode, allowing you to upload a new, functional code.

3. **The pneumatic system seems faulty.**
   Ensure that the valve controlling the air supply is open, all pneumatic cables are properly connected, and the electrovalve is in the horizontal position. If the electrovalve is not in its designated working position, it may be faulty.

4. **The robot encountered an unwanted obstacle.**
   If the motors are obstructed, a protection protocol will trigger due to an overvoltage in the circuit. Open the box next to the Delta robot and check the motor's power supply (Figure F.3). If the bulb circled in orange is lit, an overvoltage has occurred. Turn off the box or unplug its power socket, wait ten minutes, and then restore power. Normally, the bulb circled in green should light up, indicating that the issue is resolved, and you can resume operation.

Figure F.3: If the bulb circled in orange is lit, indicating an orange light, an overvoltage has occurred. If the bulb circled in green is lit, indicating a green light, the system is functioning correctly.

5. **Trouble when trying to flash a trajectory using the GUI.**
   To flash a trajectory from the interface, the system must recognize the "Teknic Clearcore" serial port, not just the "Clearcore" one, which indicates Bootloader mode. When in Bootloader mode, the interface buttons are not clickable. If a flash attempt fails and the Clearcore remains in Bootloader mode, press the reset button once, and then try flashing the trajectory again.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**