Universidade de Aveiro Departamento de Engenharia Mecânica
Departamento de Electrónica, Telecomunicações e Informática
2020

Diego Saraiva Nunes

Development of a Framework for an Automated Mechanical Testing

Desenvolvimento de um Sistema Automatizado para Ensaios Mecânicos

Universidade de Aveiro
2020

Departamento de Engenharia Mecânica
Departamento de Electrónica, Telecomunicações e Informática

Diego Saraiva Nunes

Development of a Framework for an Automated Mechanical Testing

Desenvolvimento de um Sistema Automatizado para Ensaios Mecânicos

Dissertação apresentada à Universidade de Aveiro para o cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Automação Industrial, realizada sob orientação científica de António Gil D'Orey de Andrade Campos, Professor Auxiliar c/Agregação do Departamento de Engenharia Mecânica da Universidade de Aveiro e de Miguel Armando Riem de Oliveira, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro.

O júri/ The jury

Presidente / President	**Prof. Doutor Rui Manuel Escadas Ramos Martins** Professor Auxiliar da Universidade de Aveiro
Vogais / Examiners Commitee	**Doutor Pedro André Dias Prates** Investigador e Professor Auxiliar Convidado da Faculdade de Ciências e Tecnologia da Universidade de Coimbra
	Prof. Doutor António Gil d'Orey de Andrade Campos Professor Auxiliar c/Agregação da Universidade de Aveiro (orientador)

Agradecimentos / Aknowledgements

Aproveito neste momento para tornar público o meu profundo agradecimento a todos os que, de uma maneira ou de outra, me acompanharam e contribuíram no alcance deste objetivo.
Ao professor Gil por toda a dedicação, pela energia positiva constante e acompanhamento incansável que me prestou durante todo este período, contribuindo massivamente no suporte ao longo da realização do trabalho. Agradecimento igualmente remetido ao professor Miguel Oliveira pelo grande apoio na ultrapassagem de diversos obstáculos que se foram atravessando pelo caminho e do qual sempre se disponibilizou para ajudar.
A toda a equipa OP_M9'Calibre Group por todas as sessões e contributos. Basta-me dizer que estar apenas integrado nela foi uma fonte de enorme aprendizagem e conhecimento constantemente adquirido, onde às vezes em sessões só o facto de estar calado a ouvir os temas que estavam a ser debatidos era razão mais que suficiente para estar a aprender mais sobre a área.
À empresa DigiMetrix GmbH pela amabilidade na cedência temporária das bibliotecas necessárias para a ligação ao robô.
Ao amigo Adão pela sua ajuda na sua especialidade e convívio prometido e ao Engenheiro Rui Heitor por todo o suporte logístico, companhia e inspiradoras conversas tidas no laboratório.
Um grande obrigado em especial, às minhas duas grandes fontes de inspiração, os meus pais. Sem o suporte e os princípios passados ao longo desta vida nada disto seria possível alcançar, tendo neles um exemplo constante a seguir e a razão pelo qual tento de dia para dia tornar-me numa pessoa melhor.
À minha amada Inês, pela amizade e por tomado conta do leme tanta vez para conseguirmos juntos seguir o nosso rumo. O orgulho enorme que tenho nela preenche-me o coração diariamente, e todo o amor por ela nunca será o suficiente para agradecer o que tem feito por mim.
Ao meu irmão pelo enorme exemplo, sabedoria e inigualável afinidade que sempre nos uniu. À Cristina, por se ter juntado a nós ao longo do caminho da vida e igualmente ser um grande exemplo e inspiração a seguir.
Por fim, às nossas duas meninas recém-chegadas ao mundo, Isabella e Eva, que a vida vos traga felicidade sempre. Vamos estar sempre aqui a contribuir para isso e, que o tio no futuro vos possa ler este pequeno excerto para vos motivar nos vossos percursos académicos a seguir.

Palavras-Chave Manipulação de provetes, Ensaios mecânicos, Automação, LabVIEW, Controle de robô, Calibrações, ROS, ChArUco.

Resumo

Atualmente, prever ou caracterizar o comportamento dos materiais é uma tarefa demasiado laboriosa que envolve a realização de muitos testes repetitivos utilizando diferentes tipos de máquinas de ensaio, sensores internos (células de carga para medir forças) e externos (câmaras que medem deformações). Estes ensaios, quando executados manualmente por um operador, originam erros de imprecisão devido à repetibilidade do processo. No entanto, estas tarefas podem ser adaptadas a recursos automáticos, tirando partido da fiabilidade dos manipuladores robóticos e, aumentando assim, a precisão e quantidade de ensaios realizados no mesmo espaço de tempo. Algumas soluções comerciais já existem no mercado, porém, estas são oferecidas de forma integral que não permitem facilmente a integração de uma nova estação (equipamento) ou o uso de uma máquina de ensaios standard mais antiga. Neste trabalho é criada uma estrutura de comunicação entre os equipamentos que permite a interface e, consequentemente, a reprodução automática de um sistema para ensaios mecânicos numa máquina tradicional de carregamento uniaxial. Para tal, um conjunto de alterações numa máquina de ensaios é sugerido visando aumentar o seu grau de automação e, consequentemente, contribuindo para a integração num sistema de ensaios totalmente automático. Adicionalmente, é criado um sistema de perceção visual (com as devidas calibrações em ROS) capaz proceder à manipulação de provetes a partir de um tabuleiro (posicionado aleatoriamente no espaço) e cujas manobras são operadas por um manipulador robótico e respetiva câmara instalados. Como resultado, o sistema executa o manuseamento automático de provetes para uma posição fixa que representa o local de abastecimento das máquinas de ensaios.

Keywords Specimen Handling, Automation, Material Testing, LabVIEW, Robot Control, Calibration, ROS, ChArUco.

Abstract Nowadays, predict or characterize material's behaviour are arduous tasks which involve performing repetitive tests using different testing machines and sensors (internal, such as load cells, and externals, such as cameras that measure deformations). When performed manually, they result in inaccurate data, due to the repeatability of the process. However, these handling specimen tasks could be performed automatically using traditional robotic manipulators, and creating systems based on automatic mechanical testing. Consequently, the precision of the testing results increases, as well as the automatism of the processes and the test's throughput. Some commercial products are already available on the market, but these options are offered in complete systems and cannot adapt themselves to existing equipment or retrofit systems. In this work, it is created a communication infrastructure between the equipment to be included in the automatic system designed. The whole process is managing the interface communication between devices and, consequently, composing the automatic material testing routine. This requires implementing several changes in the testing machine selected aiming to increase its degree of automation and, consequently, allow further integration in the fully automatic testing procedure. Additionally, a visual perception system is created using a specimen tray randomly positioned on a setup, which includes a camera and robotic manipulator that is automatically operating the traditional specimen handling of testing. The calibration methods are accomplished using ROS framework and the final system achieved operates in automatic mode, handling specimens from the tray prototype (designed for this work) to a fixed position previously taught to the manipulator, which represents the exact feeding position in machine tests.

Abbreviations and Nomenclature

Abbreviations

AMTS	Automated Material Testing Systems
ASTM	American Society for Testing and Materials
CEN	Comité Européen de Normalisation/ European Committee for Standardization, Documents and other appropriate data
Corp.	Corporation
DEM	Departamento de Engenharia Mecânica
DIC	Digital Image Correlation
EN	European Standards
FTP	File Transfer Protocol
IDE	Integrated Development Environment
ISO	International Organization for Standardization
JIS	Japanese Industrial Standards
LAR	Laboratório de Automação e Robótica
LEM	Laboratório de Ensaios Mecânicos
OpenCV	Open Source Computer Vision Library
PLC	Programmable Logical Controller
PXI	PCI eXtensions for Instrumentation
ROS-I	ROS Industrial
ROS	Robot Operating System
STL	Standard Triangle Language

TCP/IP	Transmission Control Protocol/Internet Protocol
URDF	Unified Robot Description Format
USM	User Socket Messaging
UTM	Universal Testing Machine
UTS	Ultimate Tensile Strength
VS	Microsoft Visual Studio

Nomenclature

ΔL	Length variation
σ	Stress
σ_0	Yield Strength
σ_u	Ultimate Tensile Strength
ε	Strain
ε_f	Strain at the fracture point
A	Cross-sectional area of the material
E	Modulus of elasticity
L	Length
P	Force applied to the material by the grips

Contents

List of Figures

List of Tables

Part I

Outline

Chapter 1

Introduction

Feed-stock materials are used by the entire engineering manufacturing sector to produce different parts, such as automotive and aerospace products, white-ware domestic products and among others [28]. These standard materials produced in the global industry are submitted to an exhaustive process of testing, in order to extract the maximum data about its behaviour when subjected to load conditions related to their final purposes [29].

1.1 Motivation

The knowledge about different material properties (strength, hardness, toughness, brittleness and others) is crucial, and even more important is to predict its behaviour under several conditions, because the components should satisfy the services and its requirements [30]. Life prediction of industrial plant, production quality control and designing specifications are some of the examples of these predictive data utilities [28].

The standard procedures for these typical mechanical tests are manual and human-dependent. The operator is responsible for controlling the tests and assuring its reproducibility, repeatability and the continuous monitoring of its results. Traditionally, one of the most crucial tasks of the process is also included in the operator's responsibilities: feeding the apparatus manually with specimens[1] and guarantee its perfect uniaxial alignment [1].

With the increase of material control requirements and with the rate on throughput tests getting even more demanding, considerable advances in the field of automatic testing systems are required. The evolution in this field can provide systems, in which robots are able to feed computer-controlled testing machines and bring another throughput to the set, collecting more reliable data results in less time [28].

The goal of this dissertation focuses on designing an automatic testing procedure, as contextualized on Figure 1.1. To handle specimens it is necessary to create a variation of today's manual testing procedures. For this reason, an automatic material testing system is designed using the industrial robotics and vision systems capabilities.

[1]A specimen is a material sample, with a geometrical shape established right from the start, and extracted from feed-stock materials.

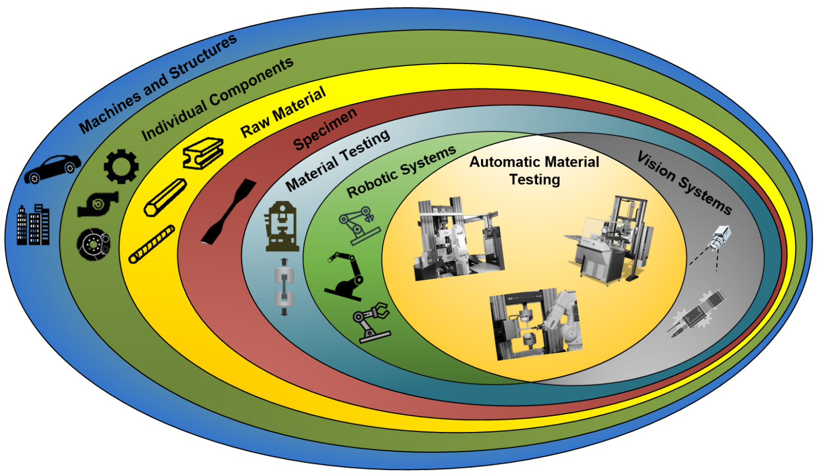


Figure 1.1: Global framework of the goal of this work.

1.2 Problems

In any testing, the data is only as good as the test methodology allows. The accuracy of a testing system depends on: (1) the quality of the testing system itself, and (2) the different errors that can be introduced into this same system along with the procedure. The fish-bone diagram, illustrated in Figure 1.2, expresses sources of errors on the results of a material test [1].

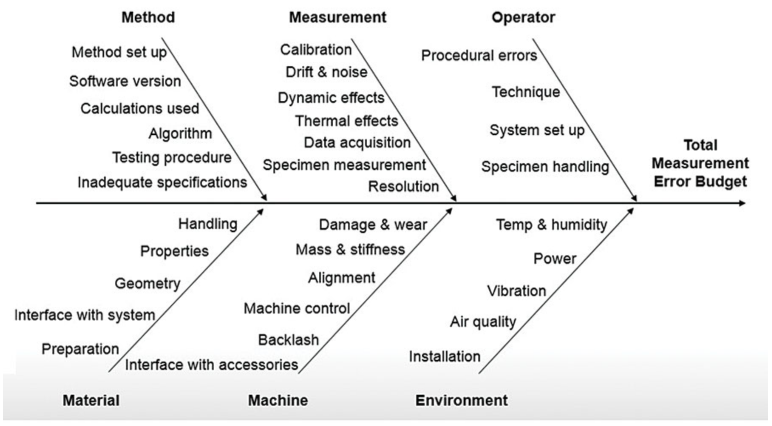


Figure 1.2: Diagram representing all the causes of errors in material testing [1].

The main focus of improvement, and the reason to use automatic systems in these procedures, is mainly the errors resulting from the operator's (listed in Figure 1.2), and the time consumption for testing. Therefore, the purpose of this section is to analyse the problems inherent to the manual standards.

The main problems are:

- **Operator handling - Reproducibility[2] problems:** In manual procedures, detecting errors from manual operators handling is problematic because these differences can only be identified by the ones who observes the testing of all operators and is trained to identify procedural errors. Each operator employs his unique technique in the process of handling or preparing the specimen, inserting the specimen into the grips or conducting tests (tasks performed by the operator conducting an experiment [10]). All these factors can reflect unintended misalignments or the lack of reliable data.
- **Repeatability[3] problems:** Executing a single test of the same material specimen, in one single testing machine does not return the consistency of results than running (by the same operator) multiple tests in the same machine in a large scale. Problems with methodology, calibrations, operators handling, material preparation, even machine maintenance and environment might appear with the repeatability procedures.
- **Supervised 24/7 systems, and skilled labour misused:** This topic also approaches problems about the skilled labour appliance. Because standard procedures are manually operated by technicians, requiring constant monitoring. These include, for example: (i) feeding the testing machine with specimens, (ii) input initial data parameters on the machine to start each test or (iii) record the data results. Unsupervised and automatic 24/7 testing is the goal. By ensuring this aim, skilled labour could be used to perform other types of technical work, in order to avoid the attribution of routine tasks to the operators. That would be an advantage to satisfy the high number of tests required.
- **Specimen homogeneity:** Complexity and quantity of the mechanical characterisation obtained depends on the number/type of tests and specimens shapes. Data about the material behaviour and parameters can significantly increase with a wide range of stresses, several strain paths and amplitudes. That can be achieved changing the specimen shape or/and the variety of the forces applied in the sample [29, 31]. The aim is to obtain a wider range of results from a unique specimen, contributing to the aim of reducing the number of tests, improving quality(precision) and time rate of each test.
- **Non-homogeneity in specimen markings:** Testing machine procedures that include Digital Image Correlation (DIC) as part of the process of results acquisition, requires a marking phase[4]. The usual method of generating a random speckle pattern is by manually spraying paint on the specimen. However, this makes it difficult to reproduce the optimal pattern for maintaining identical testing conditions and achieving consistent

[2]Reproducibly is the ability that different operators have to produce the same results from similar parts with the same level of consistency.

[3]Repeatability concerns about how well the system and process can produce known results over multiple tests, applying the same testing conditions.

[4]Usually, it consists of a random speckle pattern on the surfaces of the specimens. The speckle pattern is used to detect deformations by comparing the location of all subsets in a reference image with a sequence of images where deformation occurs.

DIC results [32].

- **Safety problems:** In non-automated material testing systems, each procedure that implies the user to operate with any machine actuators can be risky. The stages most vulnerable to risks are (i) the manual introduction of the specimen in the apparatus (alignment and centring), which requires proximity of the labourer's hands to gripping regions while these are operating, and (ii) the overall operation of the test machine because, while grips are stretching or compressing the sample, the operator's proximity could be a dangerous aspect. The remains of the specimen at the time of its fracture (e.g. in destructive tests) is an example of danger by exposure.
  In automated material testing systems, combining a complex level of sensors with actuators leads to a self-processing that requires to be as much safe as possible for some labourer around the operation. These safety levels can be increased with a simple addition of railings around the equipment or, security barriers that can prevent various hazards to the user. Safety implementation can be included at the same time in the software systems too, adding double-safety phenomena to the system.

1.3 Proposed Solution

The proposed solution intends to develop a communication framework, connecting all devices in the same infrastructure performing the role of the experiment manager, and able to automate the mechanical test procedures and its routines. It is expected to take advantage of the apparatus available on the Mechanical Engineering Department, from University of Aveiro: (i) the biaxial testing machine from the Mechanical Tests Laboratory (LEM), described by [20], and the robotic manipulator (Fanuc LR Mate 200iD [21]) from the Laboratory for Automation and Robotics (LAR). The main goal is to create an automatic routine of testing with the following features:

1. Tensile testing machine with its hardware/software adapted from the biaxial testing machine referred above, where stepper motors are actuated by Arduino UNO [33], through its respective drivers;
2. A routine established using the robotic manipulator (Fanuc LR Mate 200iD [21]) and able to pick specimens from poses returned by the visual perception algorithm, and align them in the testing region of the testing machine;
3. Visual perception algorithm returning the specimens' picking poses. It takes advantage of a camera setup (Logitech C270 Webcam [34]) pointed to a simulated specimen's tray, aiming to detect the reference marker (Aruco) and its corresponding poses;
4. Complete calibration process of the camera executed on ROS-Industrial (ROS-I) [35];
5. Main control module (in LabVIEW [36]), which combines all parts and manage the data flow of the testing.

1.4 Reading Guide

This document is grouped into seven main sections: (I) Outline, (II) State-of-the-art, (III) Intellectual Property, (IV) Practical Approach, (V) Implementations, (VI) Results, and (VII) Final Remarks.

In the Outline (I) part, the general problem and consequent motivations for this work are described, followed by the proposed solution.

In State-of-the-art (II) part, a scientific review is performed to this thematics: concepts and standard procedures incorporated on mechanical testing. An approach to robotic systems in mechanical testing is also reviewed.

In the Intellectual Property (III) part, a research on commercial and industrial patents is performed in order to evaluate the solutions available in the Automated Mechanical Testing field/market.

The Practical Approach (IV) part presents the proposed framework for the system to implement and describes the different components included in the experimental infrastructure.

The Implementations (V) part describes the tasks performed in each module of the system, and consequently, the Results part (VI) executes this system integration and presents experiments of the operation.

The conclusion and future works are reported on the Final Remarks (VII) part.

This document also includes an Appendix part (Appendix A to H) including complementary information needed throughout the work.

Part II

State-of-the-art

Chapter 2

Scientific Review

Machines, vehicles, and its structures must satisfy acceptable levels of requirements (performance, economy, safety, and durability) on the used materials. The research on the mechanical behaviour of the materials allows to understand their performance under several circumstances, in order to avoid failure in engineering applications. Excessive strength applied in a component causes deformation and damage accumulation, which may lead to failure [2].

2.1 Tests for material characterisation

The most common process in the material testing comprehends the application, in samples of the material (specimens), of different types of forces to obtain the behaviour of the given material quantitatively. Today, to guarantee robustness in engineering structures, engineers make sure that each material is tested before being used for design or production. For this reason, excessive deformations and failures caused by external or independent factors are avoided.
Basic types of material failures are classified in Figure 2.1, according to their causes.

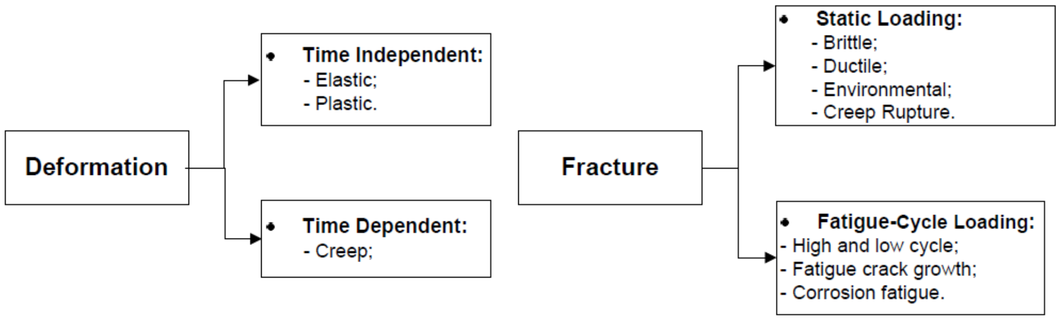


Figure 2.1: Types of deformations and fractures [2]

.

2.1.1 Deformations and Fractures

Deformations are cumulative strains in a component (bend, twist, or stretch) caused by stress, as illustrated on Figure 2.2a, where A is the cross-sectional area of the material, L and ΔL are the material length and length variation respectively, and P the force applied by the load.

In cases that stress exceeds a yield value, deformation becomes irreversible.

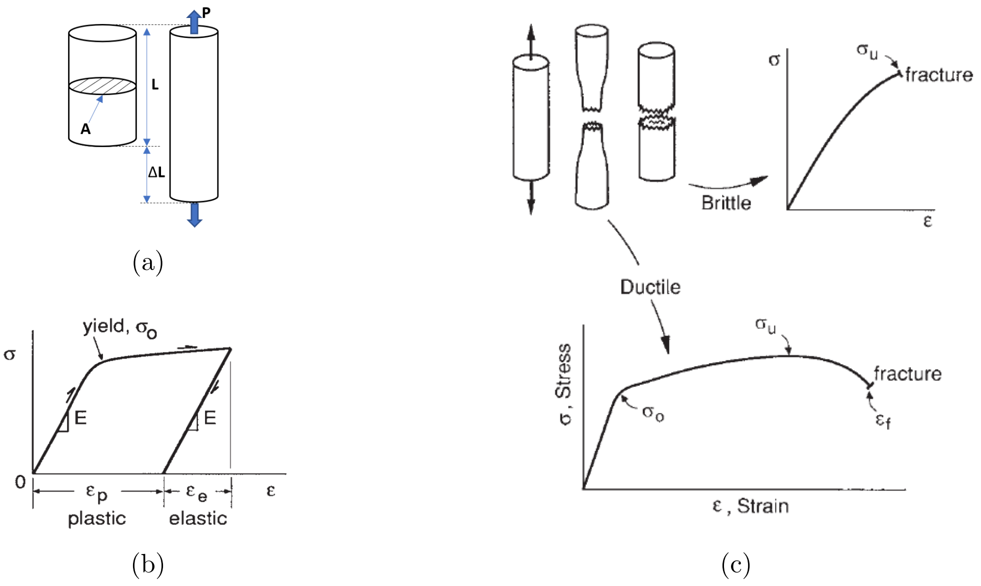


Figure 2.2: (a) Component subjected to axial loading and unloading, (b) example of both elastic and plastic deformation and (c) representation of different fracture points in ductile versus brittle material testing behaviors [2].

As a result, stress and strain of a material can be obtained depending on the relation of their physical properties and load applied to the material, being respectively given by

$$\sigma = \frac{P}{A}, \tag{2.1}$$

$$\varepsilon = \frac{\Delta L}{L}. \tag{2.2}$$

Deformation can be classified into two different categories, as illustrated in Figure 2.2b: Elastic or plastic. The elastic deformation is recovered upon unloading, while plastic deformation is permanent, not recovering upon unloading. E is the modulus of elasticity and σ_0 is the yield strength. The first is proportionally dependent, for axial loading, of stress and strain. And the second marks the transition from elastic to plastic deformation.

Another specific parameter in stress-strain curves (in mechanical tests performed by slowly stretching the sample until its fracture) is the fact that different materials behave differently according to their properties: ductile and brittle behaviour, as illustrated in Figure 2.2c. Unlike brittle behaviour, ductility in material testing is capable of sustaining a large amount of plastic deformation. σ_u or Ultimate Tensile Strength (UTS) is known as the maximum stress that a material can withstand before its fracture and ε_f is the fracture point.

2.2 Material Testing Methods

There are several material tests to analyse properties such as strength, hardness, toughness or brittleness of a specific material. These are classified into two different categories: destructive and non-destructive tests. The main component used in destructive tests is the specimen, which can be a thin sheet of the material to be tested in a desired/standard shape. The non-destructive tests are, most of them, focused on finished products.

The destructive test uses the material until a fracture point and analyses the behaviour under different loading conditions. The non-destructive test does not fracture the material, being capable of detecting internal defects of finished products [3].

According to their procedures and primary purpose, the principal tests are classified as illustrated in Figure 2.3.

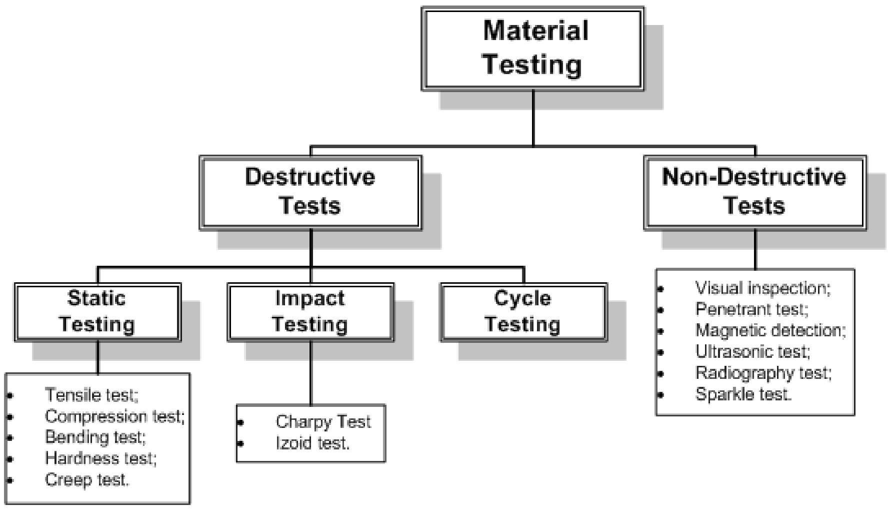


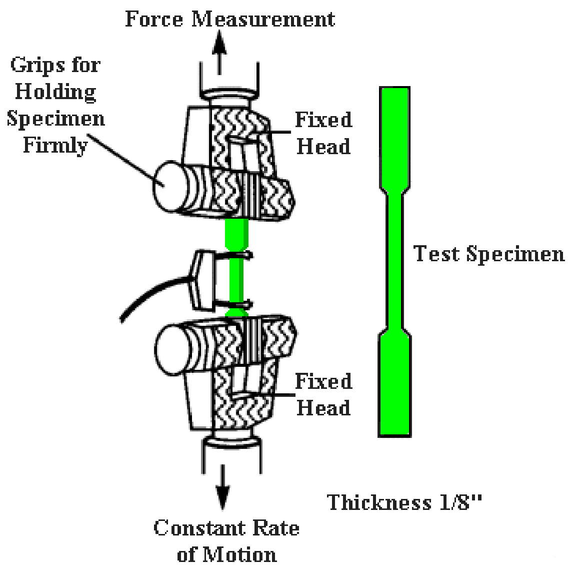
Figure 2.3: Basic types of Material Testing - Classification [3].

The mechanical testing of materials involves a wide range of testing methods and standards as ASTM, ISO or CEN [37], along with testing conditions and procedures in a variety of different environments [38].

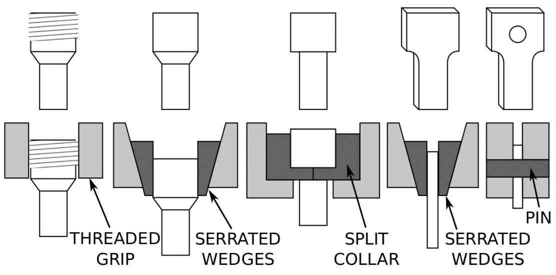
2.2.1 Tensile and Compression Tests

The tensile test is the most common test method in this field (Figure 2.4a). It provides information about tensile strengths (yield and fracture), tensile modulus and strain, and ductility of the material. The main procedure is to measure the forces required to break a specimen and the extent which stretches or elongates to that breaking point [28].

Critical points on the procedure are appointed to alignment because the materials can be anisotropic or brittle. Meaning that their properties and behaviour during the tests can be different according to the direction of the applied force or load [37], [28].



(a)



(b)

Figure 2.4: (a) Forces applied in tensile tests stretch specimens on a unidirectional axial [4] and (b) different systems for gripping tensile specimens [5].

The gripping mechanisms can include manual, pneumatic or hydraulic actuation: may be screwed into a threaded grip, or it may be pinned. Grip section may be held between wedges. The most critical concern in the selection of a gripping method is to ensure that specimen can be held at the maximum load without slippage or failure. Bending should be minimised too [5]. Based on Figure 2.4b, for round specimens, there are threaded grips and serrated wedges. For butt end specimens, split collars constrained by a solid collar. And sheet-shaped specimens can have serrated wedges or can be gripped with pins.

In compression tests, the load applied by the cross-head of the apparatus will move downwards and presses the specimen on a table. Removing the load after exceeding the specimen's elastic limit can cause fractures [39]. In general, both tests can be conducted on Universal Testing Machines (UTM) discussed in Section 2.3.

2.3 Universal Testing Machines

Testing machines are either electromechanical or hydraulic. The main difference is on the method by which the load is applied. Electromechanical apparatus is based on a variable-speed electric motor, a gear reduction system and a system of screws that move the cross-head up or down. This motion loads the specimen in tension or compression. To accurately control the speed of the cross-head, a closed-loop servo is implemented in the microprocessor system. Alternatively, hydraulic machines (illustrated in Figure 2.5) are based on a single/dual acting piston that moves the cross-head up or down. However, most static hydraulic machines have a single acting piston (or ram) controlled by the operator that adjusts the orifice of a pressure-compensated needle valve. The aim goal is to control the loading rate [5].

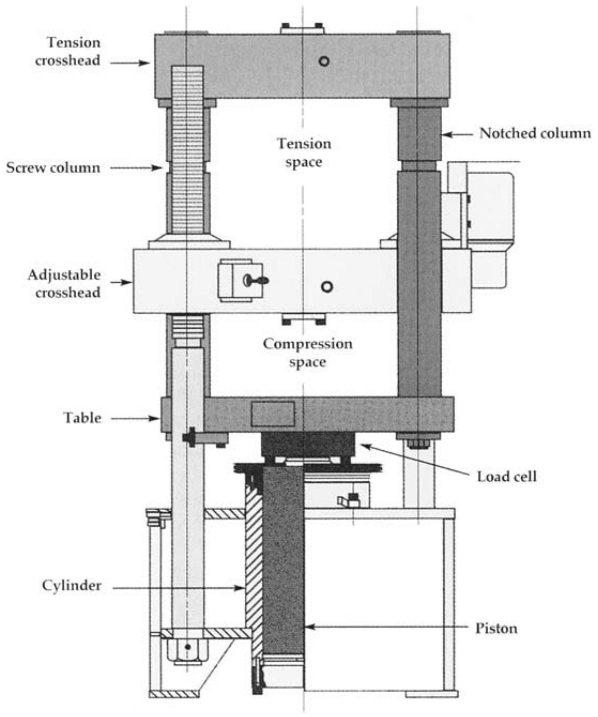


Figure 2.5: Components of an hydraulic testing machine [6].

Comparing the two systems, in general electromechanical machines are capable of a wider range of test speeds and longer cross-head displacements, whereas hydraulic machines are more cost-effective for generating higher forces [5].

2.4 Standard Procedures on Mechanical Testing

The result's accuracy through typical tests depends not only on the results outputted by the apparatus but also on the efficiency of the specimen preparation and positioning on the machine before the test. These other specimen's preparation methods (see Section 3.1) are in the majority referred to as pre-treatments or preparations of the samples before the execution of the test itself.

2.4.1 Measurements phase

Specimen measuring devices, like micrometres or callipers, can provide digital measurements, digital measurement with automatic data entry, or manual measurements. In standard procedures, this stage is usually included. In some solutions presented in Section 3, this step is executed three times to detect variations and estimate error rates on thickness measurements

[13].

2.4.2 Marking phase

The gage marks allow obtaining deformation by measuring elongation (distance) from each other. The gage marks may be made with a centre punch, or with lines scribed using a sharp or a pointed tool. Care must be taken to ensure that the gage marks, especially those made using a punch, are not deep enough to become stress raisers, which could cause the fracture to occur through them. This precaution is especially necessary when testing materials with high strength and low ductility [5]. Applications using Digital Image Correlation (DIC) metering systems (as addressed in section 2.5.2) does not require gauge marks. The procedure is to mark the specimen with random and contrasted sparkles formed by a suitable and easily detectable composite that enables a successful correlation of the image subsets generated during the process [6]. This process will allow, in real-time, accurate tracking of the testing by video or images sequences.

2.4.3 Specimen alignment and load

The worn grips or specimen misalignment are potential problems. With the physical alignment not guaranteed, any off-centre loading will exert bending loads on the specimen while testing, causing load-measurement errors due to the passage of bending forces through the load-measuring apparatus. Alignment can be affected by the testing machine load-frame, any grips and fixtures used, and also by the specimen itself [40].

2.5 Metering Systems - Contacting vs Optical devices

The measurement devices influence the accuracy of the results. Metering systems, as strain measurement devices, are traditionally divided into two categories: Contacting and Non-Contacting. Contact metering systems are carried out using some form of contacting extensometer, where a typical clip-on can be attached to the specimen with clips or elastic bands and use knife-edges to track deformation in a specimen during testing [41] accurately. Optical or Non-contacting systems, such as DIC, are analytical techniques that compare images of a specimen surface during testing to generate a full-field strain map.

2.5.1 Extensometer

The basis of this system is to evaluate the uniform deformation by measuring specimens displacement. The factor of selection is the displacement range that depends on its applications. There are two types of extensometers: stiff or dynamic and soft or static. For stiff/dynamic extensometers, precise strain measurements are obtained but with limited 5% strain measurement range. In the other hand, in soft/static extensometers, a large strain measurement is obtained but with less precision.

The general extensometer purpose is to get the average of the strain gauge length. Info about strain distribution is unknown. However, knowledge about strain distribution is essential because when local deformation occurs, tensile load initialise its peak and strain distribution is no longer uniform. The average strain becomes meaningless and useless if this phenomenon pretends to be analysed [7].

2.5.2 Digital Image Correlation - DIC

DIC technique avoids the disadvantages related to previous Section 2.5.1. The fundamentals of DIC consist of tracking a point and find the maximum correlation coefficient along with the consecutive subsets of the test. As benefits, this process gives diffuse and local necking strain (instead of constant strain) by returning a point-to-point strain field. It is an accurate and reliable approach for strain-stress curves, featuring diffuse necking.

In its setup, two cameras setup can be used to detect 3D deformations. For 2D deformations, one camera is enough. In summary, its characteristics are based on: being a non-contact measurement technique, full-field, suitable to larger deformations and strain measurements, able to measure displacement at a massive number points on the surface of the specimen and it uses standard white light source instead of a laser.

The measurement procedure includes (1) images of the object that are taken before deformation, (2) images of the object that are taken after deformation and (3) run correlation algorithm for the quantitative data evaluation in each subset.

In the differential coloured image of the specimens obtained, the brighter colours represent significant strain changes. It means that direct measurement result of the process is converted to a real strain (by numerical differential method), defined as a strain data to engineering strain conversion. This process is embedded in DIC's software [42].

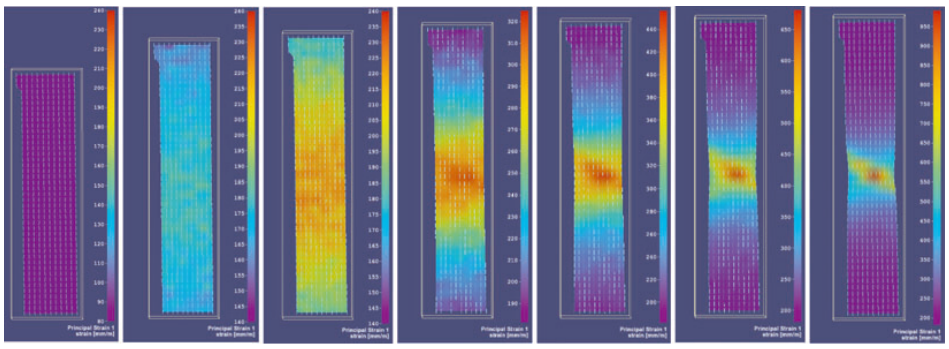


Figure 2.6: Basic types of deformations and fractures measured by DIC [7].

On [7] experiment, and according to the ASTM standards, DIC method (Figure 2.6) is applied in sheet metal tensile specimens simultaneously using a contacting extensometer measurement too. Defining A and C as edge points, B the central point and D the extensometer results, major strain vs times is obtained, as illustrated on Figure 2.7.

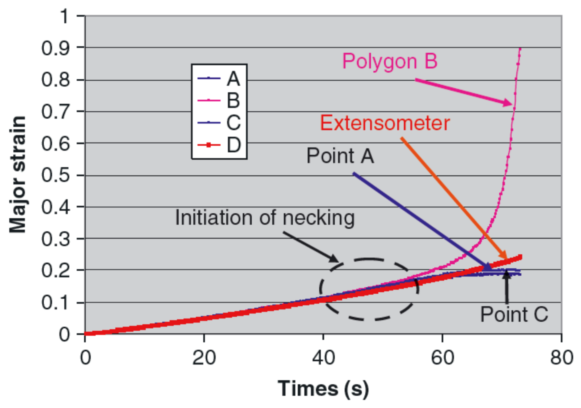


Figure 2.7: Major strain-time profile with DIC and traditional method [7].

Results show that the DIC technique enables to track the strain development over the entire specimen length, and thus helps in identifying the necking or strain concentration regions during a tensile test.

2.6 Robotic systems in mechanical testing

Due to the ongoing trend toward automation and continued technical innovations, the demands for industrial robotics have risen since 2010. In spite of the recent downturn of the global economy and the negative marks of the recent trade tensions in 2019, industries like automotive, electrical, or metal and machinery have been kept as the most important customers of industrial robotics, as shown on Figure 2.8.
Design and production departments have this new concept of using new materials to develop more energy-efficient systems-based, and they are simultaneously pushed by the high competition in all major markets. This factor demands more investment with guarantee and an improved prediction of the material's behaviour. By using the standard/manual procedures in mechanical testing these imprecisions become real, being the reason manufacturers of testing machines are following these ongoing trends toward robotic systems and design fully-automatic testing systems (see market's solutions approached on the intellectual property of Chapter 3).

In the material testing field, 40-60% portions of tests performed are repetitive tests, i.e., same test conditions applied to specimens of similar geometry [43]. Table 2.1 shows the test times for one person using different degrees of automation. The inclusion of robotic systems in material testing can provide a wide range of data, being particularly useful in large

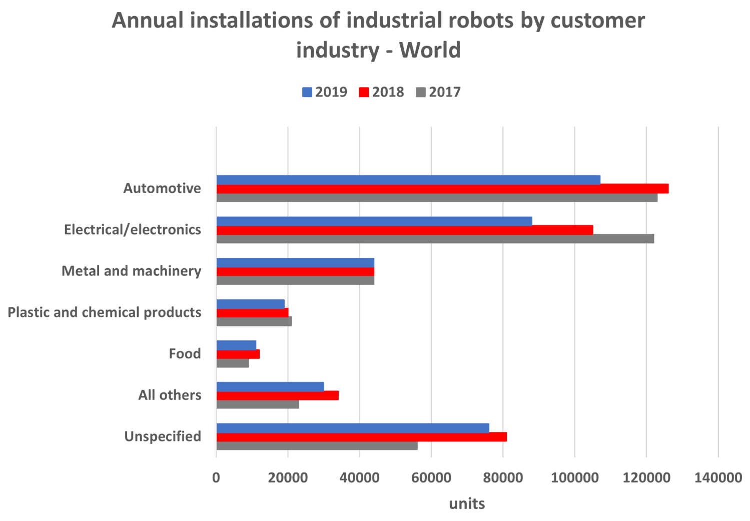


Figure 2.8: Annual installations of industrial robots at year-end worldwide by industries 2017-2019 [8].

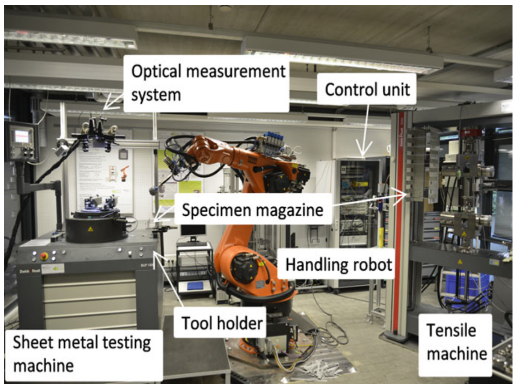
scale repetitive testing applications [44].

Table 2.1: Different degrees of automation [26].

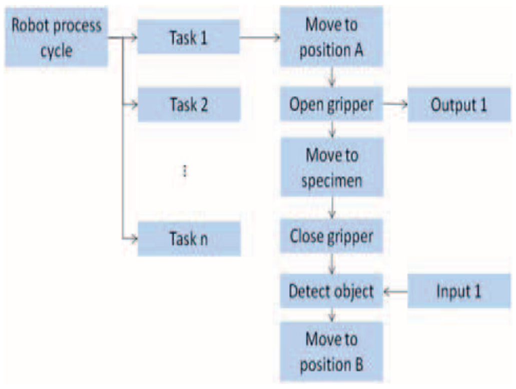
Levels of automation	**Specimens in advance**	**Time (system can operate unattended)**
Automatic tensile specimen loading	5 specimens	10-20 min
Automatic loading with larger magazines	15 specimens	30-60 min
Automatic specimen measurement	50 specimens	100-200 min
Incorporation of additional testing units such as hardness and surface roughness testing	200-400 specimens	400-600 min

2.6.1 Testing cells

The workspace definition, and the components to include in the automated process as well, are one of the first steps to be implemented. The positioning and orientation of all components have to be assigned according to the main coordinate system defined. Another type of primary requirement is to make the robot arm reachable to all intended positions. In [11], a universal and sheet metal testing machines are included within the reachable boundaries of the cell defined (Figure 2.9a), alongside with the specimens magazine and the gripper exchange stations. Sadiki et al. [10] suggests to firstly model the testing cell and the kinematic properties of all components with a CAD software as effective methods. This process allows verifying the accessibility of the robot to each component. Note that it is fundamental to select a robotic manipulator whose movements covers the spherical workspace around. In [10], an industrial robot Kuka KR 30-3 with six-degrees-of-freedom is used, due to its versatility in manipulation and for space-saving reasons.



(a)



(b)

Figure 2.9: (a) Equipment allocation in the fully automatic testing cell, and (b) flowchart of an automated process cycle, where each task is not exclusively assigned to the robot [9],[10].

2.6.2 Process cycle

In these automatic systems, each robot has corresponding tasks defined. In the remote laboratory system implemented on [10], the cyclic process is composed by a sequence of tasks as shows in Figure 2.9b, where each task is not addressed exclusively to the robot. A synchronized exchange of information between the machine and the robot itself is crucial and the actions included in each task can differ depending on the aims intended. For example, the robot can wait while the testing takes place in the machine, or can handle the next specimen to the measurement or identification stations (if existent). In [9], a sheet metal testing machine (cupping test) and a lubrification unit are included in the same testing cell of the remote laboratory referred above. There, the tasks have to be adapted in order to the handling robot perform these specific specimen feeding requirements.

2.6.3 Communication infrastructure of the automated systems

Typically, commercial solutions of testing machines require specific communication drivers to control basic functions and its associated costs to automate material testing systems. The need to contact the manufacturer's technical support for intervention is frequently implicit. Moreover, older testing machines are mainly designed for non-automatic usage. For example, having manual non-shift wedge grips does not allow to validate the automatic process immediately to implement, requiring previous adaptations/retrofit. In [9], more advanced machines are included in the cell, being already delivered with a remote interface (based on a specific communication protocol TCP/IP) and pneumatic grips, allowing the desired control.

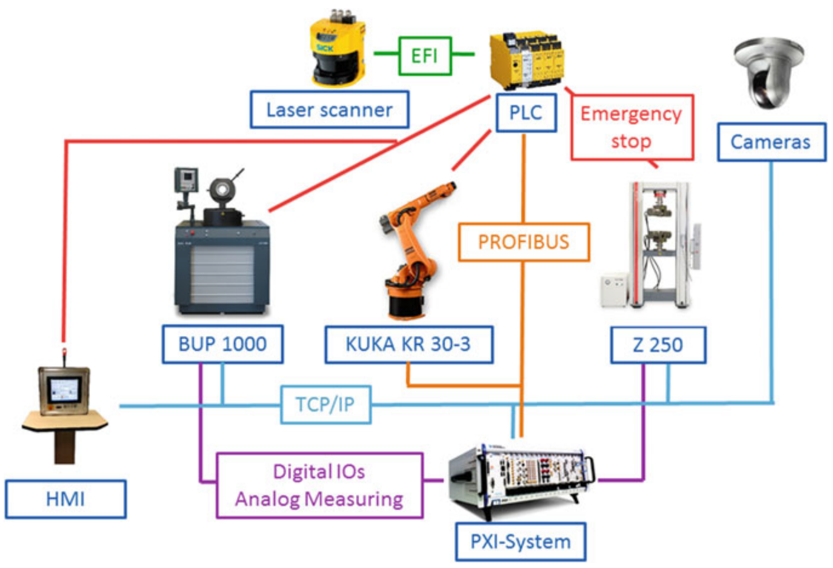


Figure 2.10: Different communication interfaces of the tele-operative testing cell created in [11].

All these factors will influence on the communication infrastructure, protocols and definition of the master controller to manage the experiment when decisions have to be made to project an automated testing system. In remote laboratories used for engineering education purposes, such as for example Sadiki et al. [10], the PXI system controller establishes different types of communication protocols with the different types of equipment, as expressed in Figure 2.10. A TCP/IP protocol is set to both testing machines, cameras, and HMI. Additionally, a Profibus protocol communication is defined to the industrial robot used (Kuka KR 30-3) and the PLC, which secures all the safety system. The controlling software, performing the role of the experiment manager, is running on the PXI-System controller with real-time support and is developed in LabVIEW [36]. Here, T.R. Ortelt et al. [11] secured the synchronization of the several routines during the experiment.

Part III

Intellectual Property

Chapter 3

Commercial Review

Due demanding requirements, it urges the development of systems capable of eliminating human influence, ensuring reproducibility and accuracy of the results, free up personnel for more complex tasks, allow overnight testing, eliminates continuous monitoring and accelerate data acquisition and processing functions [45]. For these reasons, the AMTS (Automated Material Testing Systems) concept is presented, alongside with the different stations embedded in the process.

Chapter 3 presents a state-of-the-art about these commercial solutions reviewed.

3.1 Introduction to Automated Material Testing Systems

AMTS systems have to be compliant with the demands placed on standard tests and system operations [12]. After the thorough analysis of today's procedures to execute a material test (see Chapter 2), automatic systems can be sorted in a general routine. These automatic systems have to include all required stages previously described. Considering the market solutions available, it is possible to establish a general cyclic procedure based on different stations, as illustrated in Figure 3.1.

Defining a Material Testing System as an automated process, it is necessary to guarantee the stations effectiveness. Each station has different operations as described:

- **Conveyor System - Station 1:** refers to the feeding system chosen to be part of the process: a robotic or a non-robotic (drivers/axis-based) equipment can be used to guarantee the specimens flow along with the tests.
- **Storage and Feeding System - Station 2:** refers to the way specimens are stored before the test and the process used to feed the system. For example, it can be used vertical racks or trays, and the robotic system is programmed to choose each one individually to enter in the process flow. An alternative way can be a feeding mechanism embedded on the storage system (more common on non-robotic conveyor systems). For example, a driver-based process considered as a slide feeding process inline or a carousel feeding system, where the specimens are stored on a carousel shaped magazine, making the specimens load-dependent on the circular movement of the magazine.
- **Specimen Treatment - Station 3:** includes a serial of sub-stations executing pre-treatments required on the specimen before being loaded to the test apparatus. Depending on the test, it may include for example :

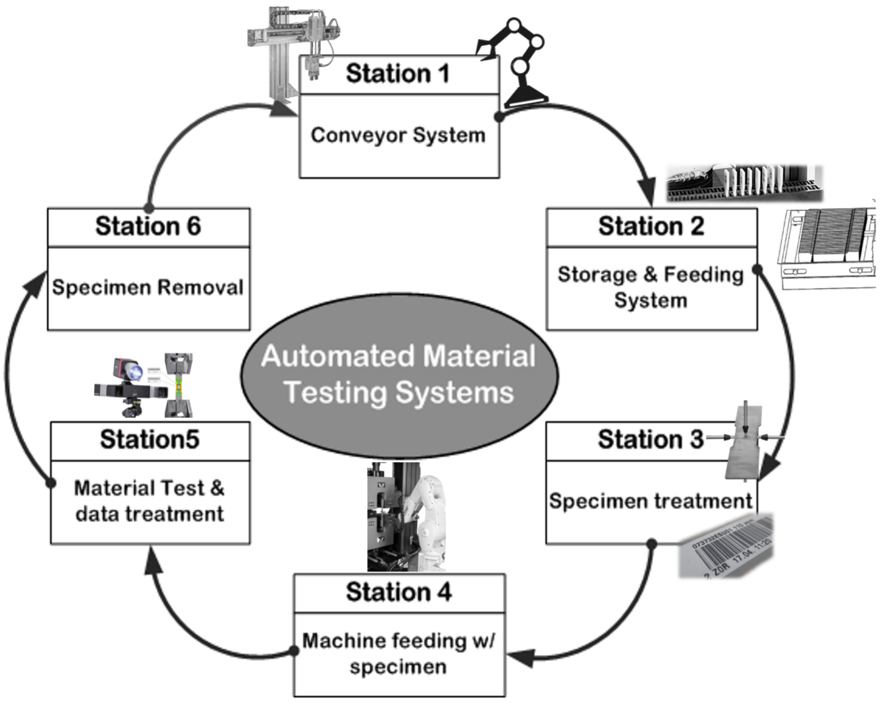


Figure 3.1: Automated Material Testing Systems stations.

- Specimen centring: more common on robotic conveyors embedded on the system. It is an indexer process done before the apparatus load and guarantees that the gripper always handles the sample in the same position with accuracy. The end-effector alight it in a station, where its mechanical shape allows samples indexation and consequent alignment. Then, the end-effector goes to pick the same sample (head-on) on a standard/programmed position;
- Specimen measurements: station-based on multiple thickness measurement devices;
- Specimen identification: bar code/QR code identification station that allows accurate tracking;
- Specimen marking for DIC measurement systems: dot, sparks or another kind of markings can be made on this sub-station to assist the metering systems to collect data;
- Specimen pre-heating: if the test intends to study the behaviour of a specimen under hot or cold conditions.

- **Specimen Positioning on the Apparatus - Station 4:**This station consists of the precise and consistent placement of specimen in the machine test grippers. Robotic conveyors assure the process and, on driver-based conveyors, indexers are used to placing the specimen always on the same location with insignificant/non-existent alignment errors.

- **Material Test and Data Treatment - Station 5:** There are two procedures embedded in this station. The classical material tests addressed in section section 2.2 and the metering systems used in this station with the purpose to collect data in real-time and obtain final results of the sample's behaviour during the test. Contacting metering systems (for example, extensometers) or Non-contacting metering systems (Video extensometer or DIC) can be used as mentioned in section section 2.5.
- **Specimen Removal - Station 6:** as a last stage of the process, station 6 refers to an automatic specimen removal of the apparatus. Numerous possibilities exist, such as aspiration/suction (plastic samples), double or individual gripper removal or a driver-based/pneumatic mechanisms designed for this purpose.

3.2 *Shimadzu Corp.* solutions

The Japanese company Shimadzu Corp. has three different systems [46], with a fully automatic procedure. Specifications are indicated in Table 3.1. The rubber tensile, the plastic tensile/bending and the tensile testing systems are available.

Table 3.1: Specifications of Shimadzu's Automatic Solutions [27].

	Rubber Tensile Testing System	**Plastic Testing Machine**		**Tensile Testing System**
		Tensile Testing	**Bending Testing**	
Applicable specimen	JIS K6251 No 3 JIS K6251 No 5	JIS K7162 ISO 527	JIS K7171 ISO 178	JIS No. 5 ASTM 1/2"
Load capacity	1 kN	10 kN	5 kN	100 kN (Machine force)
Test speed	5 to 500mm/min	0.0005 to 1000 mm/min		Unspecified
Grip, bending jig	Pneumatic eccentric roller type grip: 1kN	Pneumatic flat type: 10 kN	Bending span: 30to 150mm	Unspecified
Gauge length (exten.)	Contact type (GL20 ,25 mm)	MFN (GL50 mm)	Laser	Unspecified
Measuring device	Linear gauge type	Magnet scale type (Thickness)		Magnet scale Laser scan
Storage unit	Pallet type	Pallet or magazine type		Stack up system
Processing unit (data)	JIS K6250 K6251 K6252 (optional)	JIS K7161 ISO 527	JIS K7171 ISO 178 compliant	- Specimen size; - Upper yield; -Yield strength; -Max. stress; -Fracture elongation.

Therefore, automatic solutions available are designed for different types of material. Their basis is similar, using non-robotic mechanisms with pneumatic/hydraulic operations-based. The majority of these systems are to execute tensile tests. In Rubber Tensile Testing System, storing procedure has the concept of each sample being placed on a pallet with a limit number (8-10). Pallets are arranged in a storage device being called at the time. Stamped lines are marked on the samples to measure elongation using a video extensometer during the tests, enabling their detection. In this specific rubber system, it is adapted a thermostatic chamber where the tests take place, and the sample can be pre-heated.

3.3 *ZwickRoell* solutions

In contrast, Zwick Roell company offers a large selection of robotic testing systems, with a variety of features to suit all applications. Three different types are offered: Compact, Advanced and High-End testing systems.

3.3.1 Compact Testing Systems

Table 3.2: General specifications of RoboTest N and RobotTestC [12].

	RoboTest N	**RoboTestC**
Machine capacity	20 kN	300-600 kN
Feeding System (Magazines capacity)	20-30 specimens	24 or 40 specimen
Industry Controller	testXpert autoEdition3	testXpert autoEdition2
Applications	Simple pick and place	Tensile tests on metal specimens: DIN EN10002-1, ASTM E8,etc

Compact robotic testing systems are suitable for routine tensile or flexure tests. As benefits, testing systems with RoboTest N integrated does not require complex programming knowledge or significant hardware footprint impact than conventional industrial robots for operation due to its lightweight and force-controlled drive axis (illustrated in Figure 3.2b). It has an intuitive interface that ensures flexible use for future testing applications. Specifications are displayed in Table 3.2.

The *RoboTestC* system implemented is also considered as a compact system. Here, specimens up to 5 kg maximum in weight can be tested and, in addition, manual tests are still possible by simply pushing the automatic feeding aside when required. The system is illustrated in Figure 3.2a.

Another compact testing system, but with a different methodology is the robotic testing system RoboTest H. It is based on pendulum impact. This type of system is further being

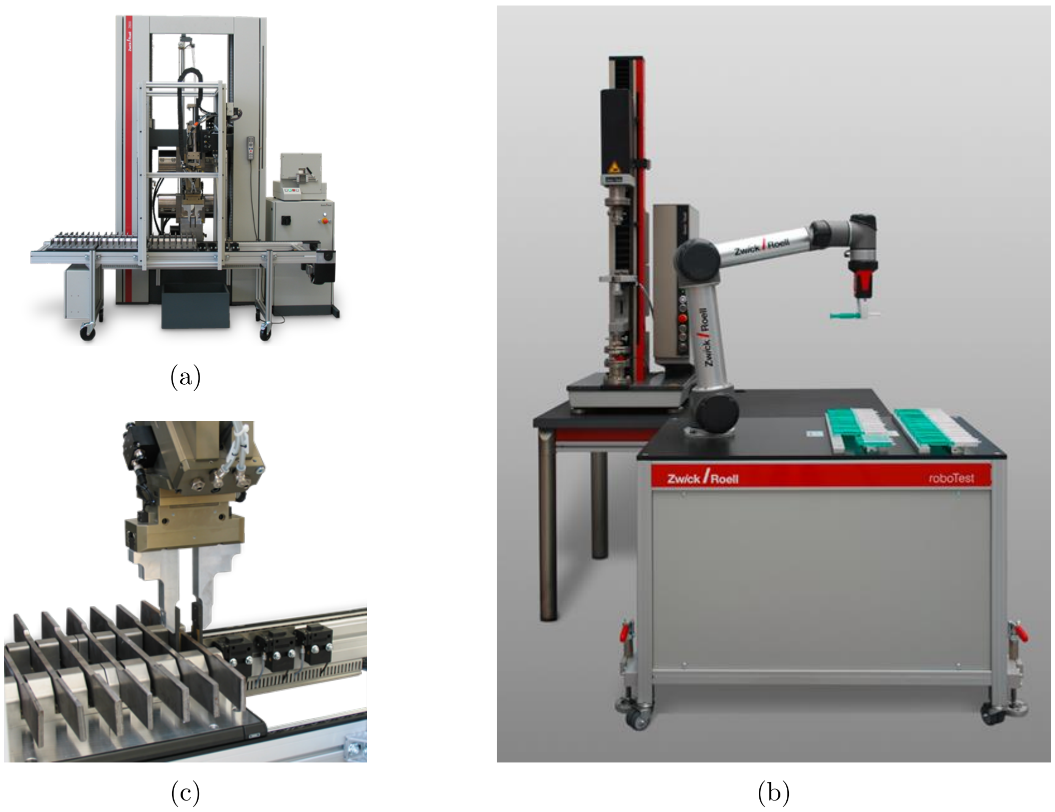


Figure 3.2: (a) Robotic testing system RoboTest C with a testing machine of 600 kN, (b) RoboTest N - The Cobot can be integrated on small test series and be processed fully automatically. (c) Pincer gripper removes a specimen from the magazine - RoboTest C [12].

compared to one similar advanced testing system (RoboTest I) in section 3.3.4.

3.3.2 Advanced Testing Systems

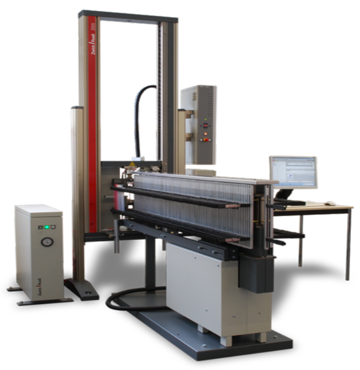
Advanced testing systems offer expanded functionality and options for tensile and flexure tests [12]. The general sequence depends on the trigger given by the operator to sort the specimens on magazines, to configure the tests and database and to start the system. After this, the system automatically executes its programmed procedure and, by stages, the robot (or driver-axis) removes the specimen from the magazine and take it to the cross-section measuring unit, composed by transducers, to read thickness and weight. Then, the end-effector insert the sample in a buffer to positioning (guarantee perfect alignment of the grips with the specimen), and the last steps are loading the testing machine and then remove it on end.

In this category, two types of advanced testing systems are analysed (i) RoboTest F and (ii) RoboTest L. These systems distinguish from each other on their feeding system and samples conveyor embedded. Typically, RroboTest F has a carousel method of feeding the

Table 3.3: RoboTest F and RobotTest L specifications [12].

	RoboTest F		**RoboTestL**
Specimen Applications	**Non-Rigid**	films, textiles	- Metals: ISO 6892, ASTM E8, JIS Z2201; - Plastics: ISO 527-2, ASTM D790,ISO 178 ;
	Rigid	metal,plastics and wires	
Machine capacity	5 to 250 kN		300-600 kN
Feeding System	Travelling axis plus a carousele with up to 200 specimen holders		Magazine holds 160 to 600 specimens
Specimen Weight	500 g maximum		1 Kg maximum
Optional extras	HOST connection		barcode reader, cross-section measuring device, HOST connection, accept/reject feature, disposal gripper, status indicator.

machine test (Figure 3.3a), and the RoboTest L has a driver axis-based system that allows picking samples from the magazine (Figure 3.3b). Feeding modules can also be used to feed specimens to a wide range of testing machines (hardness measuring instruments, notch-cutting machines) due to its flexibility.


(a)

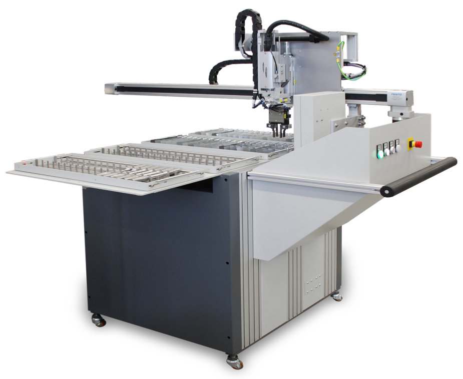

(b)

Figure 3.3: (a) RoboTest F (b) RoboTest L [12].

Moreover, RoboTest X is an advanced testing system that is characterised by offering customised solutions for special applications, such as compression testing on iron ore pellets or colour measurement transmission and reflection on plastic panels.

3.3.3 High-End Testing Systems

These systems offer the widest range of expansion options and are suitable for complex testing situations. RobotTest P is known as the systems with a portal, and the RobotTest R has a six-axis industrial robotic system included. General specifications are presented in Table 3.4.

Table 3.4: RoboTest P and RobotTest R specifications [12].

	RoboTest P	**RoboTest R**
Applicable specimens	Metals: for example ISO 6892	Metals: ISO 6892 Plastics: ISO 527-2, ISO 178, ASTM D638
Machine capacity	5 to 2000 kN	
Feeding system	Driver axis-based	6-axis industrial robot
Magazine	Up to 400 specimens	Up to 600 specimens
Specimens weight	10 kg max.	30 kg max.
Manual tests	Manual test can be executed in one of of the portals, while other is automatically testing	Robot can be put into park position
Optional extras	Barcode reader, cross-section, hardness and roughness measuring devices, temperature chamber, spectral analysis	

As shown in Figure 3.4, due to its characteristics, the portal can allow manual testing while, at the same time, runs an automatic test in other station. For this reason, a mechanical block is fixed in the unit, and the mechanical machine is software and hardware disconnected from the automation routine.

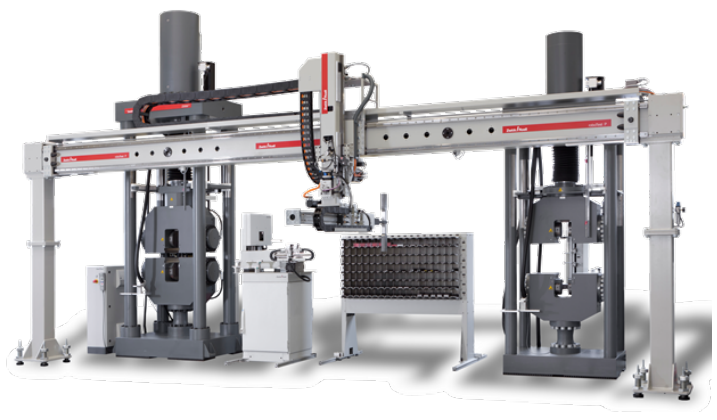


Figure 3.4: Robotic testing system - RoboTest P [12].

In Figure 3.5, it is presented the RobotTest R type system. This system has the flexibility to be used to feed specimens to a wide range of devices (e.g. pendulum impact and colourimeters).

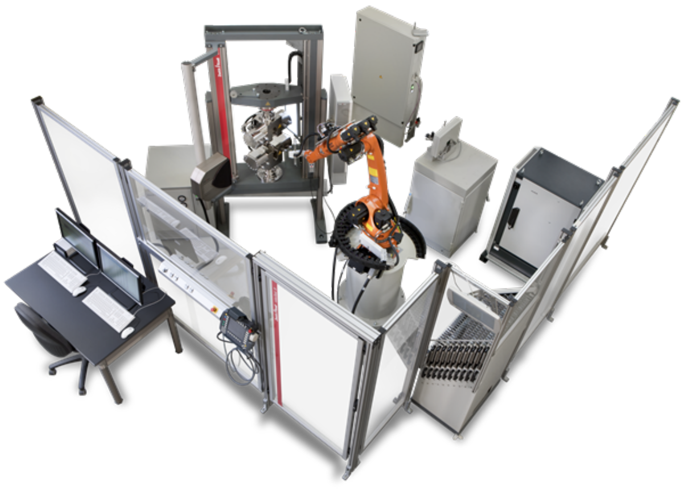


Figure 3.5: Robotic testing system RoboTest R [12].

3.3.4 RoboTest H vs RoboTest I

In this subsection, the compact testing system RoboTest H (Figure 3.6a) and the advanced testing system RobotTest I (Figure 3.6b) are presented. Both are intended to execute impact tests. General specifications are presented on Table 3.5.

Table 3.5: General specifications of impact systems - RoboTest H and RobotTest I [12].

	RoboTest H	**RoboTest I**
Applicable specimens	Charpy: ISO 179-1 Izod: ISO 180/ASTM D256	Charpy: EN10 045 or ASTM e23
Temperature conditions	Ambient temperature or cooled specimens	-180 to +300ºC
Impact energy	25J or 50J	450J (Version 1) or 750J (Version 2)
Temperature conditioning magazine	Up to 20 specimens (4mm thickness)	10-21 specimens (Version 2 has an integrated magazine holds up to 90 specimens)
Performance	5 seconds	
Specimens weight	10 kg max.	30 kg max.
Optional extras	- Manual test; - autoEdition2 software.	- Manual tests; - Barcode reader (2D); -HOST connection; - autoEdition3.

Additionally, in RoboTest I, specimen feeding is characterised by having a slide carrier insertion/removal method of the specimen into the temperature conditioning unit and then into the specimen support in machine test.

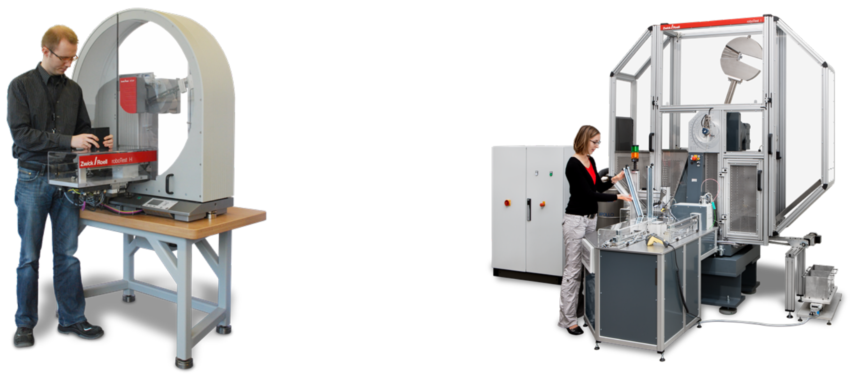

(a) (b)

Figure 3.6: (a) Robotic testing system RoboTest H and (b) RoboTest I [12].

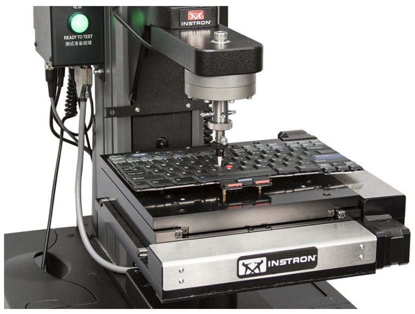
3.4 *Instron* solutions

Instron company has met the standards with XY/XY rotary stage (AT2 series) system, 3-axis non-robotic mechanisms (AT3 series) and system configuration, including a robot (AT6). They have a wide range of applications such as rubber tensile and tear, plastics tensile and flex, thin films tensile, metals tensile, and component testing.

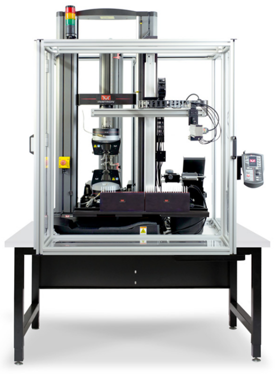
The AT2 and AT2+ automated XY and XY-rotary stage systems are ideal for compression, flex and tension testing of multiple small components. AT2+ offers an additional degree of control and efficiency, allowing the system to reach test points in different axis (illustration on Figure 3.7a). Machine load capacity is up to 2 kN, and it has a flexible interface that allows modifications of testing parameters to grow and change with products tested. Its setup has small footprint characteristics due to its dimensions. Besides, due to its rotary degree added, AT2+ is capable of executing a compression test in multi-surface components.

The AT3 series are used to perform tensile testing of plastic and other lightweight materials up to 700g weight (machines load capacity up to 50 kN), as illustrated in Figure 3.7b. AT3 model has a three-axis motor-driven and linear actuators design for automatic testing and has a smaller footprint than AT6 robotic system, resulting in a compact and straightforward solution. With an additional 90 degrees axis of motion, AT3+ model has the capability to perform automatic flexural testing.

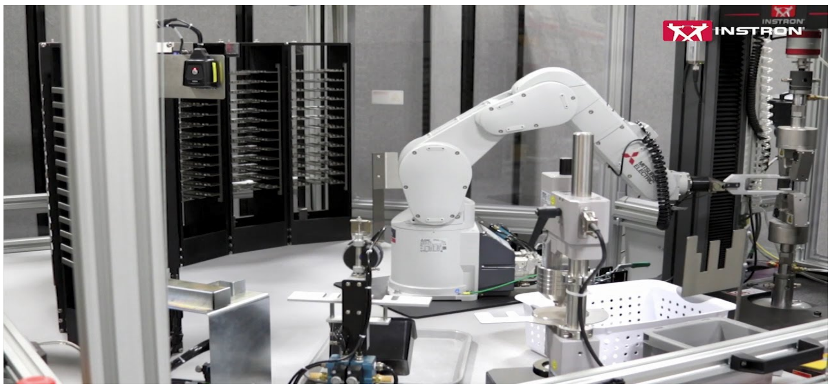
The AT6 series configuration includes a robot to guarantee the execution of specimen handling through the different steps of the process: bar-code reading, positioning on dual-axis measurement device for in-line measurement of width and thickness or test-load position. Due to its significant footprint, AT6 system has the ability to add an additional third-party peripheral device, such as a hardness station or chemical analysis station. The main features of the system are the machine load capacity up to 600 kN, robot's setup made on a roll away-table for easy conversion to manual settings or an advanced video and contacting extensometer integrated, for axial and transverse strain measurements. An example of a fully integrated system is illustrated in Figure 3.7c. The types of materials allowed to test in this system are (i) Metals (ASTM E8, EN10002-1 and ISO 6892), (ii) Plastics (ASTM D638, D790, D882 and ISO 527-2,527-3, 178), (iii) Composites (ASTM D3039 and ISO 527-4) and (iv) Elastomers or Tears (ASTM D412 and ISO 37).



(a)



(b)



(c)

Figure 3.7: (a) Automated XY AT2 system,(b) 3 axis non-robotic AT3 system and (c) Robot configuration AT6 system [13].

Generally, Instron models with a significant footprints system are completed with an enclosure to isolate the system. This factor increases safety by keeping the operator away from the testing area (decrease repetitive motion injuries, elimination of potential hazards and hands-free removal of debris) and improving ergonomics by virtually eliminating repetitive motions associated to the high volume of manual testing.

Chapter 4

Industrial Patents

The full automatic tensile testing machine reviewed in [14] comprises different modules, as illustrated in Figure 4.1: the test apparatus, the automatic extensometer, a cross-sectional measurement mechanism, a specimen tray and the pushing mechanism (conveyor) who promotes the manipulator to cross the tray station, with the specimen clamped on its grips, to the measurement and testing section.

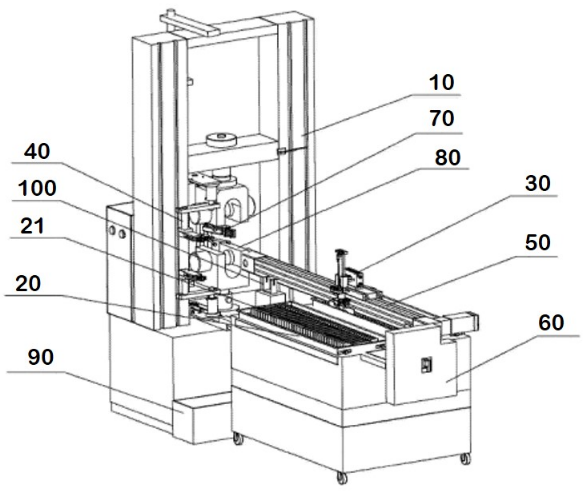
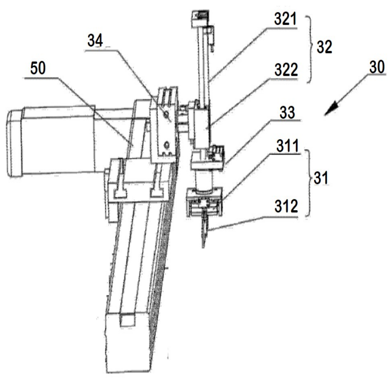


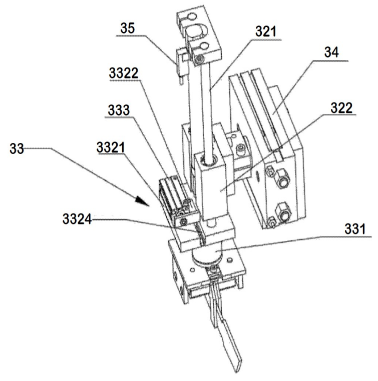
Figure 4.1: **10-** Main body of the testing machine, **20-** Specimen tray, **30**-Manipulator, **50-** Pushing mechanism, **70-** Automatic extensometer, **90-** Specimen recovery unit [14].

The working process of the system consists in using the manipulator (represented in Figure 4.2b) to pick sorted specimens from the tray by the manipulator, with its clamping mechanism. The conveyor mechanism, with driver-based specifications, is represented in Figure 4.2a with the manipulator incorporated. Next step is to transport the specimen to the cross-sectional measurement station located on the rear of the module, represented in Figure 4.1, and where measurements of the specimen are made.

In the last step, the hydraulic conveyor mechanism pushes the manipulator, with the specimen clamped on its grips, to the module's front (machine test section). Here, the degrees of freedom of the manipulator guarantee the alignment and positioning of the specimen in the testing section to allow the machine's pneumatic/hydraulic wedge grips to attach the specimen and begin the traditional testing on the machine. Finished the test, the specimen is transported to the specimen recovery unit.



(a) **31**- Clamping mechanism,
50- Hydraulic conveyor.



(b) **33**-Horizontal rotating mechanism (±90°),
34- Cylinder rotary-type mechanism.

Figure 4.2: (a) Conveyor mechanism + manipulator; (b) Manipulator [14].

As represented in Figure 4.2b, the importance of the two rotary components of the manipulator established by two different cylinders along the process is crucial because it guarantees the specimen handling in horizontal and vertical orientations.

The tray-shaped model reviewed in [15] refers to a specimen-placing device (Storage System- Station 2 on Section 3.1). Figure 4.3 represents the front view of the device. It has two sets of pallets, and its purpose consists of sorting the specimens, properly spaced from each other, along with the tray in suitable positions to allow clamping by the automatic manipulator. The length regulators allow to adjust the slots described in Figure 4.3 according to the specimen's total length to be tested.

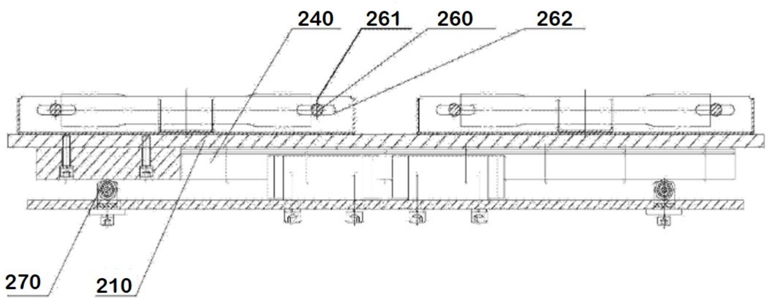


Figure 4.3: Specimen-placing device. **260**- Slot to regulate the length [15].

As a machine testing unloading mechanism or a specimen removal solution, the model reviewed in [16] presents a dual-clamp mechanism able to be integrated on the machine test-host (Station 6- Specimen Removal on Section 3.1). Symmetrical clamps, but in a different level, have the purpose to automatically remove the upper and lower remains of the specimen after the test. Illustration of the automatic system is presented in Figure 4.4.

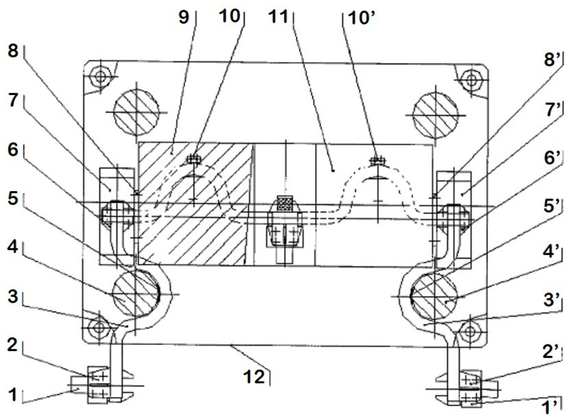


Figure 4.4: Automatic clamping and discharging device [16]:
1, 1'- Clamp grips; **4, 4'**- Index Columns; **7, 7'**- Cylinders with rotating shafts; **12, 12'**- Host machine testing.

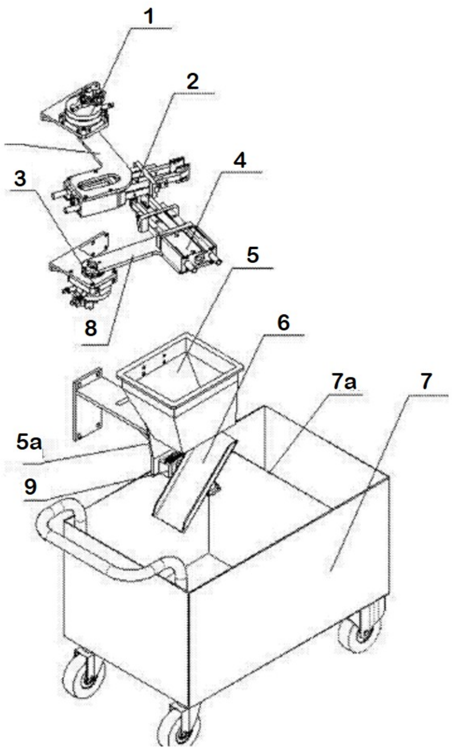
Rotating shafts on the cylinders allow the grippers to clamp the upper and lower remains of the specimen on the testing section and discharge the machine to allow the next test.

The automatic device described in [17], including the integrated part, referred on [18], is another example of unloading specimens mechanism too. Its principle of work is based on two arms comprised of both rotating shaft cylinders too (Figure 4.5a) with the same function to unload the machine test of specimens after the testing.

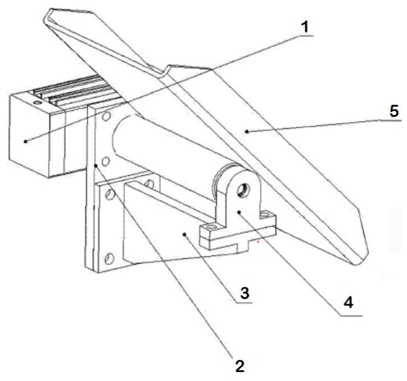
In addition, this mechanism is comprised of a funnel, a turnover plate (illustrated in Figure 4.5b) and a container to collect different remains of the specimens. There can be taken out on the upper and lower clamps of the tensile test machine, and then put into the corresponding sample collector according to the computer-controlled program. The sorting device of [18] provides a faster operation by rotary selecting the direction of the different compartments to sort the specimen.

To a different type of specimen material, another automatic feeder and fibre tester device is analysed in [19]. This system uses a manipulator as a conveyor, as represented in 4.6a.

The feeding platform operation is guaranteed by its stepper motor and carousel movement. The system is demonstrated in Figure 4.6b. The platform's function is to provide the fibre to the end-effector of the manipulator. This manipulator has a rotational cylinder axis to allow its cylindrical movements. The manipulator's end, with the specimen clamped on its end-effector, supplies the tensile fibre tester with fibre specimen, positioning them on the upper and lower grip of the machine tester.

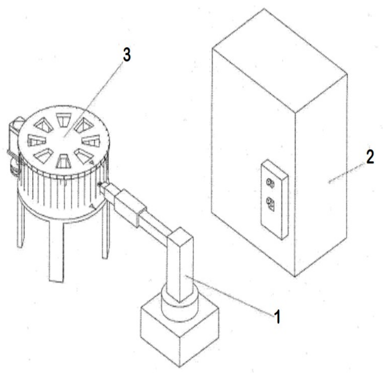


(a) Automatic sample unloading device:
1, 3-Air cylinders with rotary axis;
2, 4-Air gripping cylinders;
8-Unloading mechanical arms;
5- Funnel;
6- Turnover plate;
7- Sample collector.

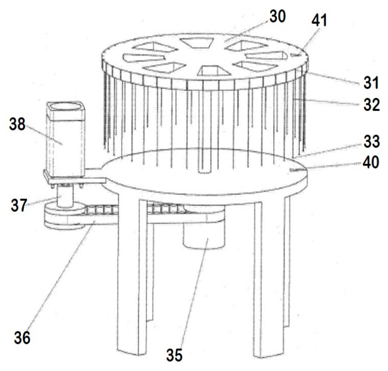


(b) Sorting device:
1- Rotating air cylinder;
5- Overturning plate.

Figure 4.5: Automatic device to discharge specimens from tensile testing machines [17, 18].



(a) Different units of the automated system:
1- Manipulator;
2- Tensile fiber tester;
3- Feeding platform.



(b) Feeding platform:
32- Fiber;
38- Stepper Motor

Figure 4.6: Automatic tensile testing system for fiber specimens [19]

Part IV

Practical Approach

Chapter 5

Experimental Infrastructure

The automated system proposed in this work is presented in Figure 5.1. The different modules required for full testing procedure are represented, as well as the communication framework between them. The central element is the Control Module which, throughout LabVIEW software, manages the complete communication between the different parts. In addition to the Control Module, the other three parts developed for this work are (1) the Actuator Component (included on the testing machine module), (2) the Visual Perception and, (3) Robotic Modules.

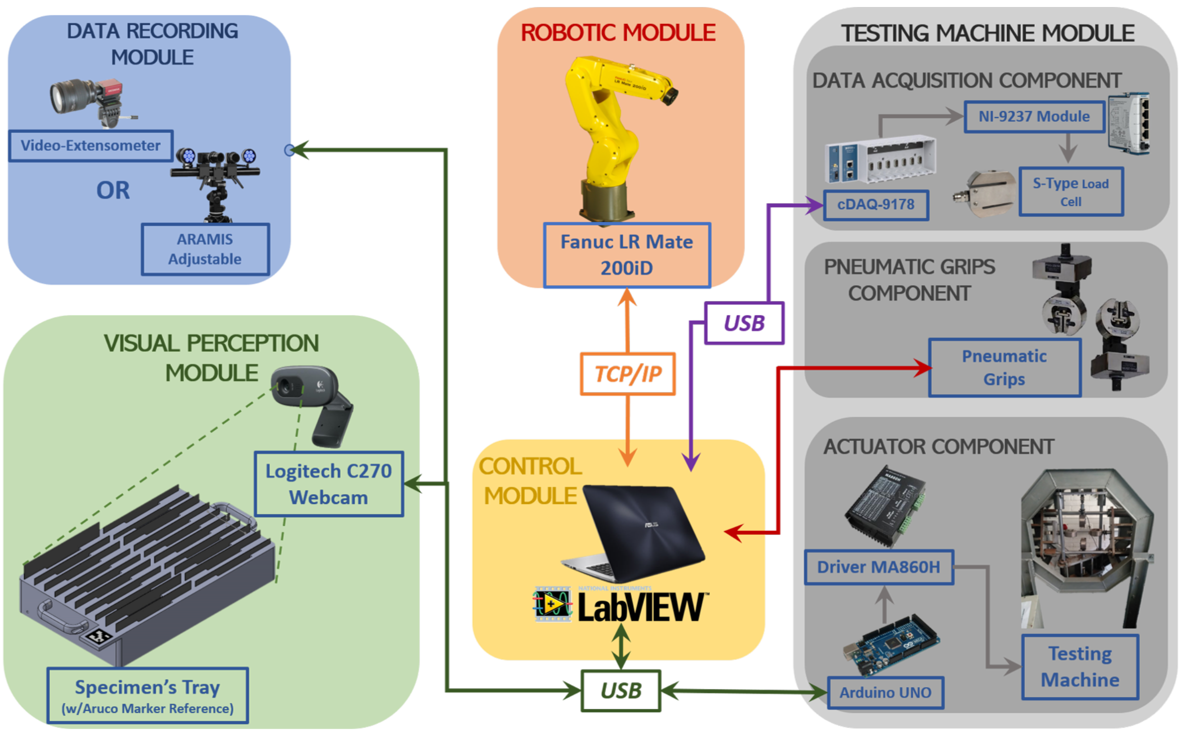


Figure 5.1: Communication framework of the automated testing system.

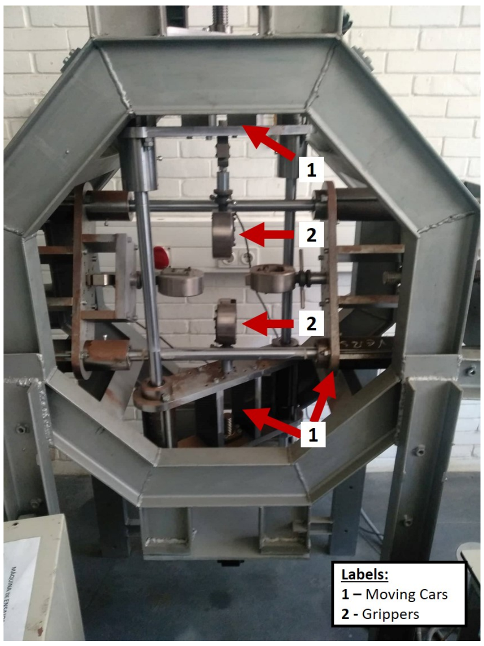
The following sections describe the hardware and software required to implement within the different modules.

5.1 Hardware

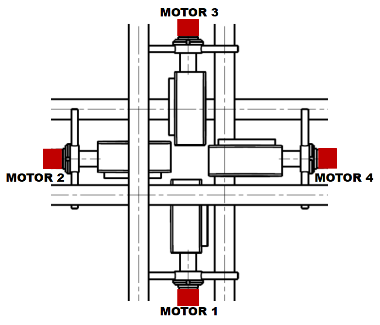
In the Controle Module, the PC used for launching LabVIEW and integrate all components is an Intel(R) Core(TM) i5-7200, with a CPU Base Frequency of 2.50 GHz, 2 Cores and 4 Logical Processors. The other hardware integrated on the system is detailed in the sections below.

5.1.1 Testing Machine

It is required a testing machine able to perform the most common type of mechanical tests in this field: tensile testing. The best solution available is the biaxial testing machine (presented in Figure 5.2a), built for the Mechanical Testing Laboratory (LEM) of the Mechanical Engineering Department (DEM). In [20], the grippers are disposed on a cruciform geometry, following the concept of biaxial testing. This mechanism has four stepper motors attached to each gripper and is controlled by the digital outputs of an Arduino UNO [33]. Each different trigger states of the outputs are transferred to the motor drivers (MA860H [47]), allowing their operation.



(a)



(b)

(c)

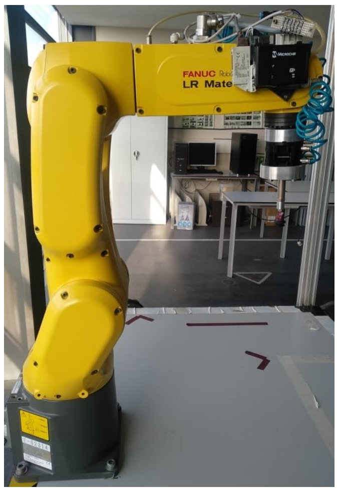
Figure 5.2: (a) Biaxial testing machine selected for the experiment, (b) the layout of its stepper motors, and (c) its manual Shimadzu 5kN grip pair [20].

In this work, to recreate the tensile testing operation, three stepper motors need to be disabled (while testing), according to the layout presented in Figure 5.2b: (i) horizontally, the motors number 2 and 4, and, (ii) vertically, the motor number 1. The upper motor 3 is the only enabled on the Control Module to perform the uniaxial testing direction.

The machine's selection criteria are supported on the easy access to the communication drivers of the apparatus. These drivers allow the implementation of control commands on the machine. Initially, the most suitable choice on the lab was the Shimadzu AG-X plus 100 kN [48]. The common obstacle of both evaluated is their mechanical grips (see Figure 5.2c) that does not allow the remote control required for automated testing[1] (manual operation). Therefore, in the decision-making process, the machine which returns the communication and control in the most open-source way is the first biaxial referred, being necessary some hardware/software adaptations to perform the automatic tensile testing.

5.1.2 Robotic System

For the Robotic Module of the automated testing system framework, the manipulator selected is the Fanuc LR Mate 200iD [21], from the Automation and Robotics Lab (LAR). This robot has the approximate reach and size of a human arm (reaches 717 mm) and has 7.00kg of payload. Its specifical datasheet is available on Appendix B. The six controlled axis provide high versatility in tasks fulfilment, and it is suitable to perform the required specimen handling referred for this work.



(a)

(b)

Figure 5.3: (a) Fanuc LR Mate 200iD [21], and (b) R-30iB Mate controller [22].

The robot is paired with an R-30iB Mate controller [22]. This component is responsible for commanding the complete hardware of the manipulator. The teach pendant guarantees the user interface by allowing several types of actions: jogging, programing, verify errors and

[1]See the request quote on hydraulic non-shift wedge grips [49] for the Shimadzu testing machine on Appendix A (the pneumatic solution [50] is 5.000€ cheaper). Moreover, the acquisition costs of its LabVIEW communication drivers to allow control commands are significant (requested quote on Appendix A).

I/O states. The original communication protocols of the controller are: (i) the server/client Hypertext Transfer Protocol (HTTP), and File Transfer Protocol (FTP). The HTTP links the PC (which sends the instructions) and the controller. The instructions received are interpreted by the server (controller), within the protocol standards, and executes on manipulator the specific tasks ordered. On the other hand, FTP protocol is used to transfer pre-compiled files from the PC to the controller directly.

In Cancela's dissertation [51], a server to the Fanuc RJ3iC controller was developed. The server was created on a client-server architecture, based on Ethernet and TCP/IP sockets. This allows connecting the Fanuc manipulator to a PC, enabling to perform the typical instructions on the industrial manipulator. In this work, the communication protocol is the same. Therefore, two FTP files from Cancela's work were executed (as the base to the TCP/IP connection) in order to configure the server.

5.1.3 Camera Setup and Specimen Tray Demo

This hardware is included in the Visual Perception Module of the framework created for the automated testing framework(see Figure 5.1). Its setup aims to detect the reference of the specimens tray (i.e., position and orientation), and the respective grasping points of each specimen. The procedure of detecting the origin of the tray is performed by Charuco's detection corners method (see below Section 5.2.5). Additionally, the algorithm has to be prepared to load a file with the fixed distances of each specimen grasping region to the coordinate reference system of the tray (demo presented on Figure 5.4b). In further works, tray designers must extract these coordinates, from their tray CAD projects, for a text file. The inclusion of this marker detectors on the tray (Aruco, Charuco) is a requirement and have to be taken into account on the components designed.

In this work, a Logitech Webcam C270 (Figure 5.4a) is used to acquire the images and a demo of a specimen tray with the Charuco markers referred (see Appendix E for technical information about the created tray demo). The simulated grasping positions of each specimen in the board are defined in an arbitrary method, but with their known coordinates (displacement) to the Charuco central coordinate system.

5.2 Software

5.2.1 LabVIEW

The LabVIEW 2020 [36] software is a system-design and development environment for visual programming in order to perform tests, measurements, and control with rapid access to hardware and data insights. In this work, the different modules integrated on the system are implemented on a graphical environment aiming to:

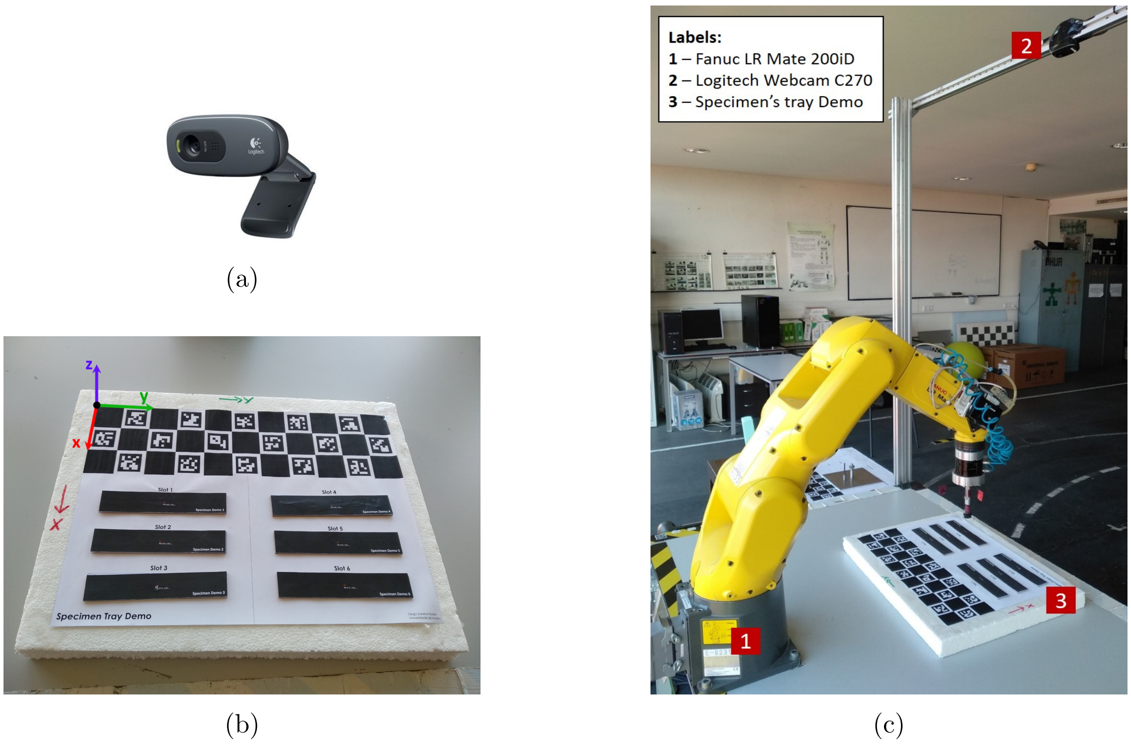


Figure 5.4: (a) Logitech Webcam C270, (b) specimen tray demo with its origin marked (referential), and (c) hardware setup with robot.

1. In Actuator Component (from Testing Machine Module), to send instructions to Arduino in order to control the motor drivers operation of the testing machine;
2. In Robotic module, to establish the command sequence for the manipulator performs the specimen handling required;
3. Visual Perception module – This node loads the file returned by the C++ code compiled in Visual Studio, which contains the converted world coordinates (robot base frame) of the specimen's grasping points returned by the *Charuco detection* algorithm.

5.2.2 Arduino IDE

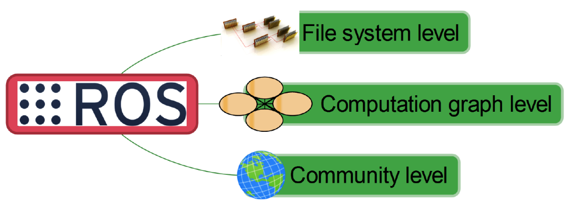
This micro-controller manages the operation of the testing machine's motors, and its respective signals sent by its digital outputs to the motor drivers. In the machine design [20], an Arduino Uno R3 has the previously mentioned role, having a USB 2.0-USB B type of connection with the PC. It is programmed by an open-source IDE (Integrated Development Environment) using C language.

For this work, the original code developed for the biaxial testing machine is adapted, in order to allow the correct operation of the aligned motors to perform vertical/uniaxial testing. Other routines developed in this work and added to the original code are hardware/software switch limit verifications (for safety reasons), and the creation of a "home" routine for each time the machine is plugged to allow position monitoring of the stepper motors.

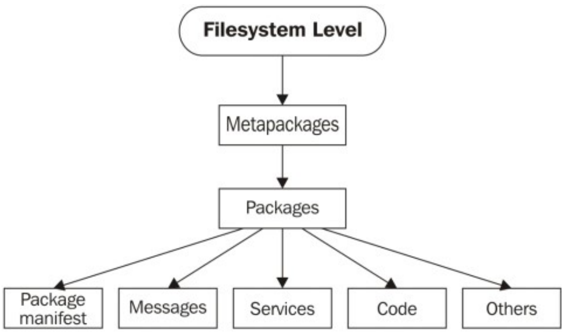
5.2.3 Robot Operating System (ROS)

In order to obtain a robust calibration with consequent high accuracy in the specimen grasping points of the tray, the complete camera calibration process (intrinsic and extrinsic parameters) is performed through ROS[2] (Robot Operating System).

The platform used in this work is ROS Melodic Morenia[3] in Ubuntu 18.04 (Bionic) release. This system is developed by the Open Robotics Foundation [54] and provides an open-source framework. At its core, ROS is not an operating system, but a Linux-based application with a collection of tools, libraries and conventions aiming to simplify the creation of complex and robust robot behaviours across a wide variety of robotic platforms [55]. The basic principle is to run a significant number of executables from different packages (if required) and enable them to exchange data synchronously or asynchronously. As expressed in Figure 5.5a, the ROS is divided into three groups: File system, Computation graph, and Community levels. The File system level defines (i) the internal operation, (ii) the storage, (iii) file structure, and (iv) the minimum requirements for its operation. At this level, the details of messages, services, and folders are specified. The standard folders that belong to ROS packages form the typical structure represented in Figure 5.5b. Therefore, it can be observed that different groups of packages are the core of ROS applications. The meta-packages can create them to process and execute specified tasks.



(a)



(b)

Figure 5.5: (a) General structure of ROS and (b) ROS File system level [23].

The Computation Graph level performs the communication between processes and the system. At this level, a network responsible for maintaining all interconnected architecture is created by ROS, enabling each ROS node interaction, access and data transfer to other nodes.

[2]Another reason for this choice is also dependent on the participation in the ROS Workshop [52] on February 2020, freely available to the group of dissertation students from LAR, and the personal will to apply in this context the knowledge acquired. The beginners' tutorials were completed apriori to help consolidate the learning [53]

[3]`http://wiki.ros.org/melodic`

The complete ROS Graph level structure, Community level and other fundamental concepts in this field were accessed on [23]. The ROS master is an indispensable key to the entire architecture management. Triggering its functionality allows the communication between nodes, establishing the exchange of information between each ROS node. The nodes communication model in the ROS environment and interaction with the ROS master is presented in Figure 5.6.

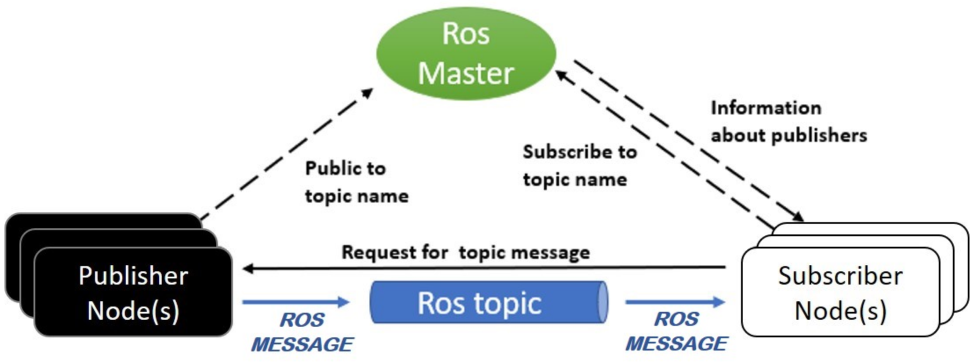


Figure 5.6: Node communication model in the ROS environment [24].

The ROS nodes communicate with each other using messages with a predefined and known data structure (standard or manually created). Each message is transported through topics method. Additionally, when the ROS node subscribes to a topic to receive a message is considered the Subscriber node. On the other hand, when the ROS node accesses to a topic name to publish its information is considered the Publisher node. This form of data transmission is unidirectional, allowing data flow on the network created without feedback [24].

ROS Industrial (ROS-I)

This software platform[4] builds upon ROS. This extension includes libraries, tools and drivers to communicate with industrial hardware. Figure 5.7 shows a representation of the packages that are organized on the top of the ROS Industrial.

Next, a brief description of the current packages within the ROS Industrial block diagram is presented:

- **ROS GUI and Layer** – The first one includes all graphical interface tools based on plugins (e.g. RViz or Gazebo), and the second one is the base layer where all communications are executed.
- **MoveIt!**[5] – offer solutions to plan trajectories, calculate inverse and direct kinematics

[4]`http://wiki.ros.org/Industrial`
[5]`https://moveit.ros.org`

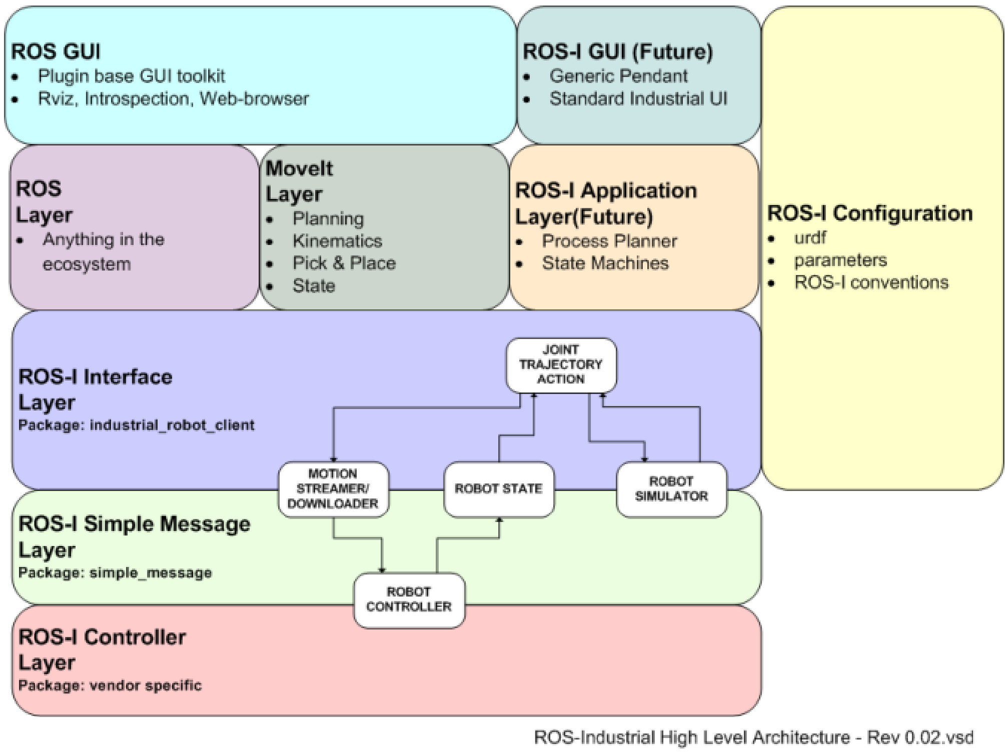


Figure 5.7: ROS-Industrial High Level Architecture [23].

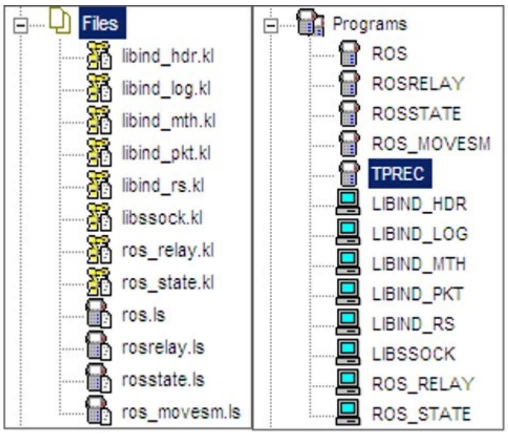
and pick and place solutions in the RViz interface.

- **ROS-I interface layer** – Consists of Robot's client. Establish a connection to the manipulator controller, using the simple message protocol.
- **ROS-I simple message** – is the communication layer with the industrial robot. It is a standard protocol that exchanges data from the client to the controller and vice-versa.
- **ROS-I controller** – is a dedicated and proprietary unit of manufacturer information of the robot.
- **ROS-I configuration** - defines parameters and design of the industrial manipulator to integrate.

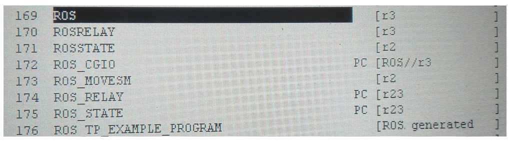
For this work, the interface of FANUC LR Mate 200iD with ROS-I is necessary for extrinsic calibration purposes. Mainly, it is necessary to obtain its current joint states and transformation frames, in several positions, that will contribute to the correlation between robot and camera frames needed. Therefore, ROS-I depends on several extra metapackages as ROS *MoveIt!*, ROS FANUC and the drivers required to install in the controller in order to be able to perform this task. The last parameter referred (drivers package) is described next section.

ROS FANUC (drivers)

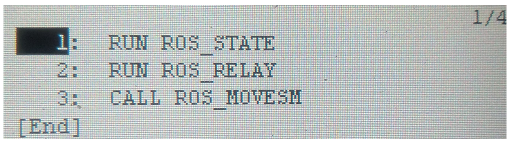
This section presents the procedure required to establish communication between the Fanuc controller and ROS-I (communication drivers). The tutorials available at [56] are followed as an aid. They describe the installation of the ROS-I server on the Fanuc controller, using the *fanuc_driver* package [57], which contains the required drivers to allow the communication. This procedure is divided into three steps: installation, configuration, and execution. Before the upload and installation of the ROS-I programs (included on *fanuc_driver* package) on the robot controller, these need to be compiled into binaries. Therefore, they are imported and compiled in the simulation software supplied by Fanuc, Roboguide[6]. Here, the executable files to send to the controller are generated (as listed in Figure 5.8a). The initial step of the installation is now ready to be executed. In this phase, the compiled files are transferred to the Fanuc controller. There are two methods to accomplish it: (i) FTP connection, or (ii) via a mass storage device. The second method is chosen by using a USB stick. Technically, this method is defined as the binary installation.


```
Files
  libind_hdr.kl
  libind_log.kl
  libind_mth.kl
  libind_pkt.kl
  libind_rs.kl
  libssock.kl
  ros_relay.kl
  ros_state.kl
  ros.ls
  rosrelay.ls
  rosstate.ls
  ros_movesm.ls
```
```
Programs
  ROS
  ROSRELAY
  ROSSTATE
  ROS_MOVESM
  TPREC
  LIBIND_HDR
  LIBIND_LOG
  LIBIND_MTH
  LIBIND_PKT
  LIBIND_RS
  LIBSSOCK
  ROS_RELAY
  ROS_STATE
```

(a)


```
169  ROS                                [r3             ]
170  ROSRELAY                           [r3             ]
171  ROSSTATE                           [r2             ]
172  ROS_CGIO                        PC [ROS//r3        ]
173  ROS_MOVESM                         [r2             ]
174  ROS_RELAY                       PC [r23            ]
175  ROS_STATE                       PC [r23            ]
176  ROS_TP_EXAMPLE_PROGRAM             [ROS generated  ]
```

(b)

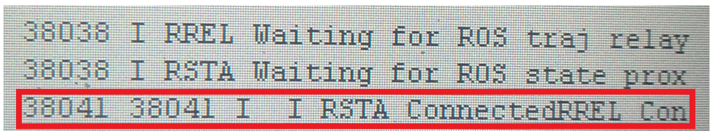

```
                                   1/4
    1:  RUN ROS_STATE
    2:  RUN ROS_RELAY
    3:  CALL ROS_MOVESM
[End]
```

(c)

Figure 5.8: (a) Fanuc driver files imported and compiled (in Roboguide), and from the teach pendant display, (b) the list of ROS programs sent to the Fanuc controller, and (c) the content of ROS program (main).

Then, the following task is the configuration of the server and the transference of the files to the controller. The configuration of the server requires two FTP files (from Cancela [51], see detailed section 5.1.2), and two specific USM (User Socket Messaging) files to create the ROS server. The Appendix C details all the ROS Server configurations performed on the controller via teach pendant. As the last step, the starting/executing of the ROS server is performed using the *ROS* program (see Figure 5.8b). Within it, the programs described in Figure 5.8c are executed simultaneously (see Appendix C).

[6]`https://bit.ly/2VPubNJ`



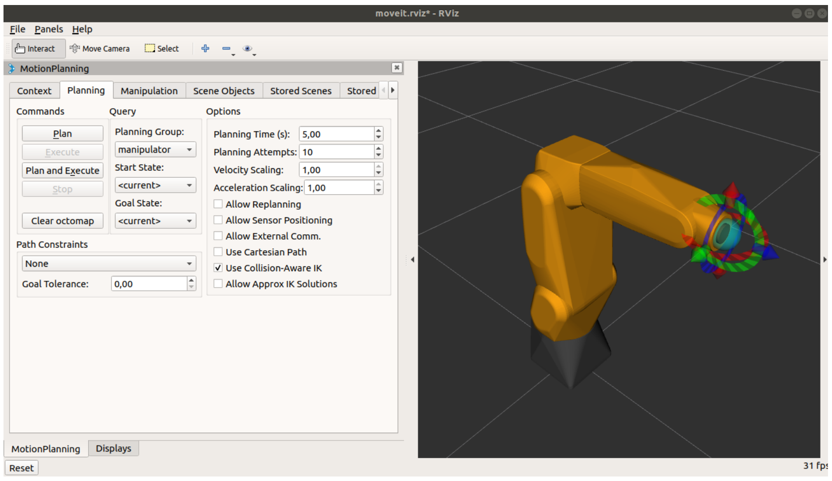
```
38038 I RREL Waiting for ROS traj relay
38038 I RSTA Waiting for ROS state prox
38041 38041 I   I RSTA ConnectedRREL Con
```

Figure 5.9: Teach pendant display after starting the server and waiting for client final connection (red).

After the execution of the *ROS* program, the server remains active and waiting for the connection of a client (see Figure 5.9). When the ROS application is executed in PC, the connection is established (red rectangle), and it waits to receive message instructions (e.g. move robot, joint states).

ROS FANUC LR Mate 200iD packages

There are three experimental packages for Fanuc LR Mate 200iD manipulator within ROS-Industrial [58]. They are responsible to establish the interactions and control, having *Moveit!* [59] platform as an assistant for planning robot's trajectories (see Figure 5.10). Therefore, all the software configurations able to control the manipulator referred meet the requirements. In this packages are included all the configuration files to the hardware, kinematic algorithms, *STL* and *URDF* models, among others.



Planning Group: manipulator
Start State: <current>
Goal State: <current>
Path Constraints: None
Goal Tolerance: 0,00
Planning Time (s): 5,00
Planning Attempts: 10
Velocity Scaling: 1,00
Acceleration Scaling: 1,00
☐ Allow Replanning
☐ Allow Sensor Positioning
☐ Allow External Comm.
☐ Use Cartesian Path
✓ Use Collision-Aware IK
☐ Allow Approx IK Solutions

Figure 5.10: Interface RViz using the *MoveIt*! library and where the Fanuc LR Mate 200iD is launched.

With all these configurations achieved, the Fanuc manipulator can be controlled by initializing the ROS Server on the teach pendant controller (run *ROS* program) and, on the PC, by the execution of the command presented next:

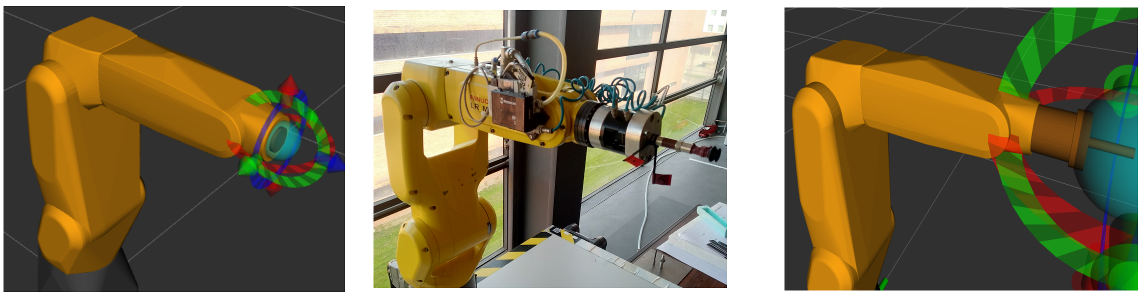
```
roslaunch fanuc_lrmate200id_moveit_config_moveit_planning_execution.launch sim:=false
    robot_ip:=192.168.0.231
```

In the parameters section, the robot IP must be selected (192.168.0.231), as well as disabling of the simulation model and consequent activation of the operation mode desired with the real manipulator by typing setting $sim := false$ parametrization. Note that, the interface with the manipulator only succeeds if the control unit has ROS, ROS-I, Fanuc drivers, ROS Fanuc LR Mate 200ID and $MoveIt!$ libraries/packages correctly installed and functioning.

URDF end-effector model

The URDF (Unified Robotic Description Format) is an XML file which contains all the visual elements of the robot (models) to enable display, collision checking, or dynamic path planning. In industrial robots, the last link always corresponds to the part which guarantees the interactions with the surrounding environment. In this case, the manipulator's model launched in *MoveIt!* does not includes end-effector (as seen in Figure 5.11a). The tool used as end-effector is replaceable depending on the needs, as well as its simulated model. Consequently, the original model in Rviz does not incorporate the gripper.

The real end-effector, coupled to the manipulator, is represented in Figure 5.11b. Therefore, it is required to update the manipulator endpoint (virtual TCP) and insert the end-effector STL model in ROS environment, in order to have coincident positions between virtual and real Tool Center Points (TCP).



(a) (b) (c)

Figure 5.11: (a) Original model of the robot, (b) real end-effector, and (c) Result including suction grip model.

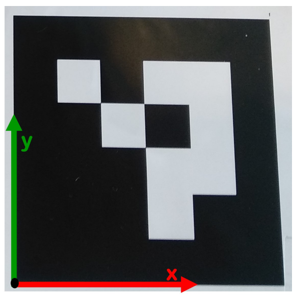
The real end-effector is measured, and a STL model with the same dimensions is created using a CAD software (*SolidWorks* [60]), being converted to STL format afterwards. This model is added to both collision and visual folders, within $fanuc\_lrmate200id\_support$ package. After that, the complete procedure to configure this tool (*tool0*) is performed through $MoveiIt!SetupAssistant$ [61] and following instructions on Simões work [62] in his Appendix A, where this full procedure is detailed. The final result is presented on the Figure 5.11c.

5.2.4 Microsoft Visual C++

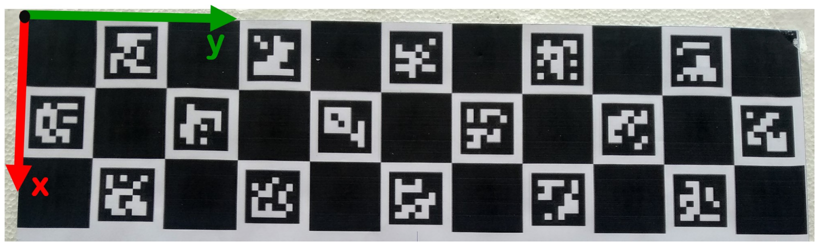
The Microsoft Visual Studio Community 2019 [63] is the platform selected to run the developed algorithm. It aims to detect the Charuco board marker (as the reference) and returns the grasping points of each specimen (in world coordinates) to LabVIEW, enabling to perform the picking task by the Fanuc manipulator. This type of detection requires Aruco libraries and the development of a computer vision application. Therefore, the Open Source Computer Vision Library (OpenCV) is used in the project. The main code is included on a new project created in a Console Application (in C++ environment) of the Visual Studio. Before the programming, the new project environment requires the installation and configuration of the OpenCV libraries. For that, the opencv_master [64] and opencv_contrib [65] repositories are used. The complete configuration process of these libraries is presented in Appendix D.

5.2.5 Detection with Aruco and Charuco Markers

Aruco is an open-source library from OpenCV for camera pose[7] estimation using squared markers. It uses a C++ language base, being versatile and with high-speed detection. The main benefit of these markers is that a single marker detection provides enough correspondences (its four corners) to obtain the camera pose [66]. Another type is Charuco markers, which combine the benefits of both checkerboard and Aruco markers [67]. In this work, two different approaches using these two different types of markers are performed (i) Aruco marker detection for Hand-Eye calibration (extrinsic parameters), and (ii) Charuco markers for determination of tray's position in the workspace.



(a)



(b)

Figure 5.12: (a) Aruco Marker used on calibration tool, and (b) Charuco board on specimen tray.

Next, the Aruco and Charuco parameters defined/selected to each marker used are detailed. The characteristics chosen for the Aruco marker applied on the calibrator tool (in Hand-Eye calibration) are:

[7]In free space, is composed of position (Point in x,y,z) and orientation (in Quarternion)

- Square size - $80mm$;
- Aruco dictionary - $DICT\_4 \times 4\_250$.

The parameters of the Charuco board selected for the specimen tray are:

- Size - 3×11 squares;
- Aruco dictionary - $DICT\_6 \times 6\_250$;
- Square Size - $40mm$;
- Aruco Size - $30mm$.

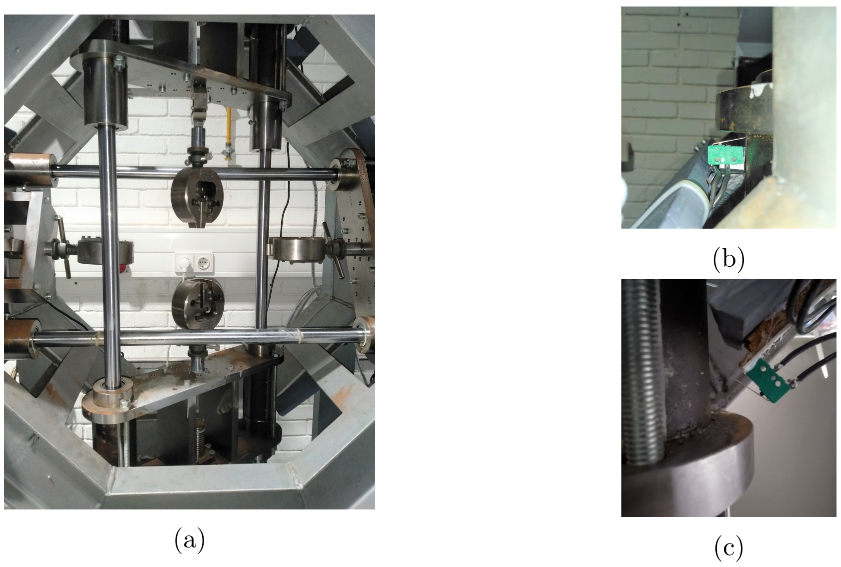
Part V

Implementations

Chapter 6

Testing Machine Module

This chapter presents the hardware/software changes performed on the converted tensile testing machine. The new features increase the level of automation in the material testing procedure. Future hydraulic/pneumatic grips to be acquired to the machine will allow remote control. During their coupling, they must be positioned with 45 degrees from the current position of Maio's [20] work, and with their openings (side where specimen is inserted) in the same direction (as illustrated in the demonstration with the mechanical grips on Figure 6.1a).


(a)

(b)

(c)

Figure 6.1: (a) Vertical grips with 45-degrees rotation, (b) Upper limit switch, and (c) Lower limit switch .

This layout enables, on one side of the machine, enough space to the robot arm to insert specimens and, on the opposite side and in line with the grips, to install a recording device while testing occurs (e.g. Video-Extensometer, Aramis). Also, other required features for automatic control are implemented in this work and described next sections: (i) limit switches to complement the function of the mechanical stopper, (ii) homing stepper motor routine, and (iii) motor positioning.

6.1 Setting up limit switches

For safety reasons, the creation of a home routine and motor positioning are features required in automated procedures. For that purpose, an upper and lower limit switches are installed on each physical stopper of the apparatus, aiming to control the courses of motors 1 and 3. Each ball screw, coupled to the motor, causes the car's motion [20]. In the original project, the mechanical limiters were installed on the machine to prevent courses leaving their desired range of motion. These electrical devices intend to be reached before the mechanical protection and can be configured/controlled throughout the software. The KW4-3z-3 125V/250V 5A limit switches are connected in Normally Open (NO) contact to the available digital ports of the Arduino (Port 10 and 11). Figure 6.2 represents the limit switches connections appended to the original electrical project.

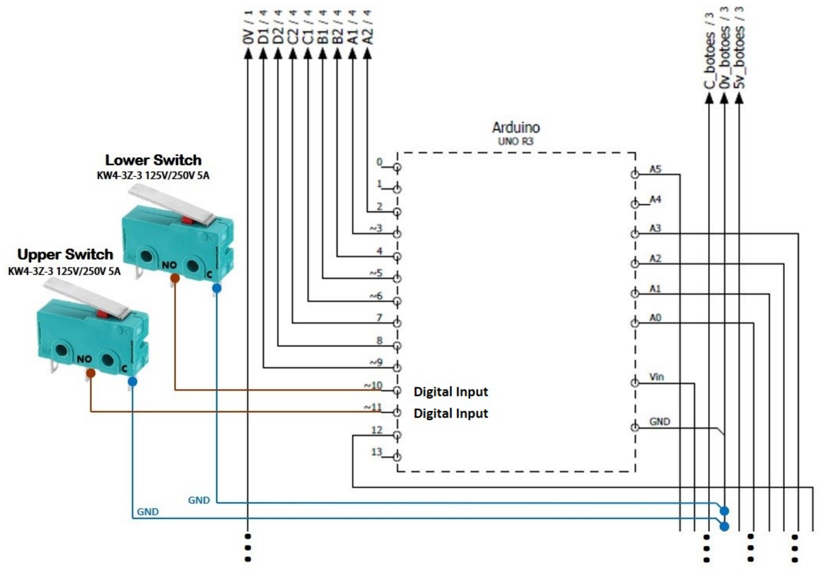


Figure 6.2: Limit switches connections added to Arduino's digital ports on the electrical project.

6.2 Homing stepper motors

Setting the "home" procedure to stepper motors requires to define limit switches as position references. The pin assignment in MA860H drivers, for both motors 1 and 3, is represented in Figure 6.3 (schematic created from the original electrical project [20]). The pulse signal (*PUL+*) allows controlling the stepper motor motion/velocity. DIR signal (*DIR+*) defines the two directions of motor rotation. The Arduino controls this pin modes by setting its digital outputs status as OUTPUT (*HIGH/LOW* or 5V/0V) and send their signals to the inputs of the pin drivers. The digital pins aimed for limit switches are set in *INPUT_PULLUP* mode.

It means that when the switches are not pressed, the internal pull-up resistors connects to 5 volts. Consequently, the Arduino pin reports *HIGH* status continuously.

In software, the homing routine is appended to the original code (in Arduino IDE environment), within the *void setup()* section. Here, every time the machine is started, the homing routine runs once until the cars reach their references (obtained by the trigger of each limit switches states).

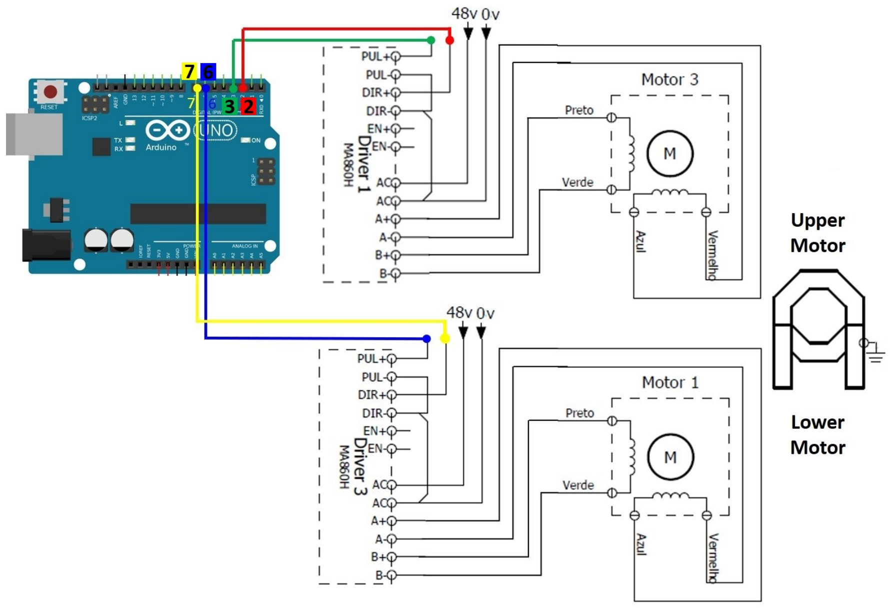


Figure 6.3: Arduino pinout connections to the MA860H drivers for tensile testing [20].

The homing loop code (sector 7 of Appendix F code) is based on the concept that the limit switches are activated while not pressed (within limit courses), and not activated when pressed (reaches limit course).

6.3 Motor positioning

After running "home" procedures, the reset to both positioning variables are performed (*pos_inf*=0 and *pos_sup*=0 on Appendix F). After that, the maximum extent course for the cars coupled to the stepper motors can be set. This positions can also be defined as the maximum extent in tensile testing or, from a mechanical perspective, the maximum extent of the ball screw coupled to the motor shafts. Through manual measurements, 100 mm from the home position is the maximum extent defined for the inferior grip, and 85 mm for the superior. Therefore, a condition to guarantee the courses within these limits is added to cover the full main code section (see Appendix F, line 77).

Both motor drivers have the original configurations set of (i) 400 steps/revolution and (ii) provide 1.8 A current to the motors. The Arduino does not configure the motor's velocity directly. The concept to control velocities with micro-controllers relies on the configuration of time *delay* between each pulse states (*HIGH/LOW*). Consequently, the relation between the car velocity (V_{carro}) and motor velocities is obtained throughout the ball screw steps and the gear reducer. This relation is given on [20] by

$$delay[\mu s] = \frac{18750}{\mathrm{V_{carro}}[\mathrm{Kmm/min}]}. \tag{6.1}$$

The main code on Arduino runs in a loop. For each iteration, the *PUL+* state is triggered on motor drivers with a time instance equal to *delay*. Tracking the current position of the grip requires to obtain the displacement that value corresponds to each pulse, and increment/decrement of that distance value in each iteration, while motors are operating. On Maio's [20] manual and automatic mode of the machine, the velocity of the cars is defined by the user (in manual mode by an analogic reader and through LabVIEW by automatic mode). Then, displacement per pulse ($dist_{pulse}$) is equal to

$$\mathrm{dist_{pulse}} = \mathrm{V_{carro}}\frac{[\mathrm{mm}]}{[\mathrm{min}]} \times \mathrm{delay}[\mu \mathrm{s}] = \frac{d_{\mathrm{carro}}[\mathrm{mm}] \times \mathrm{delay}[\mu \mathrm{s}]}{1[\mathrm{min}] * 60 \times 10^6[\mu \mathrm{s}]}, \tag{6.2}$$

where $V_{\mathrm{carro}}[\mathrm{mm/min}] = \frac{d_{\mathrm{carro}}[\mathrm{mm}]}{1[\mathrm{min}]}$. Therefore, for each velocity given by the user interface, a distance per pulse iterates/decrements (on *pos_inf* or *pos_inf* variables) in each loop of the code while the motors operate. The code listed below presents the method of distance per pulse calculation for motor 1 (in manual mode) and the increment/decrement position depending on the testing mode selected.

```
velo= analogRead(vel); // Velocity
vm1=  analogRead(m1); // Manual mode - motor 1 ON/OFF
vm3=  analogRead(m3); // Manual mode - motor 3 ON/OFF

temp= map(velo,0,1023,2000,200); // 0-1023 delay between 200 and 2000
// Formula dist_pulse defined: (dcarro[mm]*delay[us]/1min)
dist_pulse= velo*temp/(60*10^6);

// Temsile move
if (vm1>100 && vm1<639){
  digitalWrite(md1, LOW); // md1 - pin direction
  sm1=true;
  pos_inf = pos_inf + dist_pulse; // Increment dist_pulse(position) obtain in each pulse
  }
// Compress move
else if (vm1>700){
  digitalWrite(md1, HIGH); // md1 - pin direction
  sm1=true;
  pos_inf = pos_inf - dist_pulse; // decrement dist_pulse(position) obtain in each pulse
  }
```

This procedure is appended to the code to both stepper motors (1 and 3), in manual and automatic mode.

Chapter 7

Visual Perception Module

This chapter presents the features which guarantee the correct Visual Perception Module of the system operation: (i) Calibrations, and (ii) *Charucodetection* algorithm. For the proper functioning, all the required coordinate systems are attributed to each reference, aiming to calculate the relations between them. Their positions and orientations are expressed in Figure 7.1.

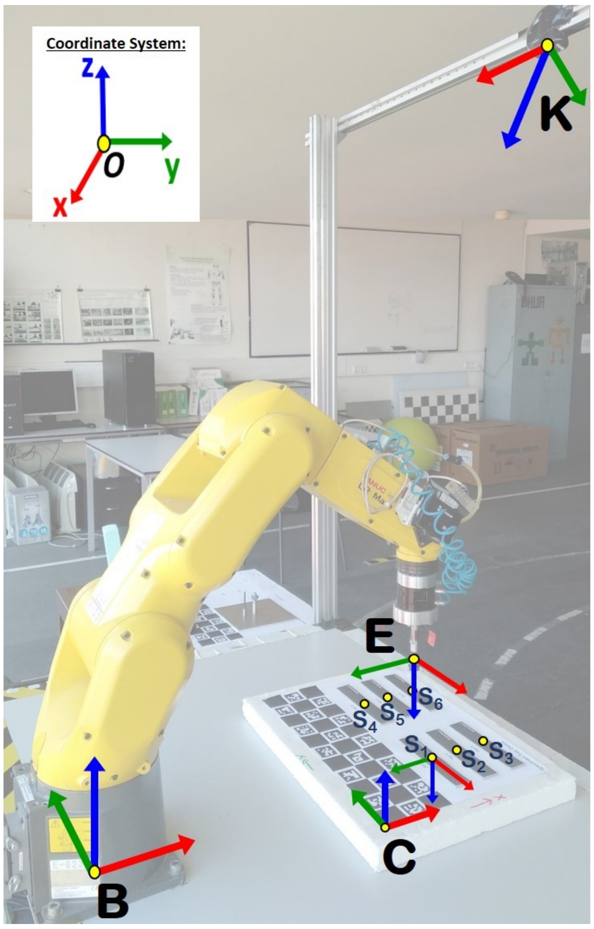


Figure 7.1: Main coordinate systems required for the Visual Perception Module.

The coordinate systems here allocated are defined as:

- **B** - Coordinate system of Fanuc's base (or *base_link* in ROS model) coinciding with the table's surface (absolute reference);
- **E** - Coordinate system of the end-effector (or *tool0* in ROS model). Current position depends on the manipulator's joint states;

- **K** - Coordinate system of the camera (fixed);
- $\mathbf{S}_n$ - Coordinate systems of n specimen[1] grasping points on the different tray slots. They coincide with the central point of each specimen to grasp, and their current pose depends on the tray's random position;
- **C** - Coordinate system of the tray's reference, and also considered as the Charuco marker reference.Similarly to $\mathbf{S}_n$, its location depends also on where the trays is placed (random position, but reachable to the robot arm and camera's FOV[2]);
- **A** - Coordinate system of the Aruco marker (used on Hand-Eye calibration detailed on Section 7.1.2).

The main goal of this module is to obtain the coordinates for each specimen grasping region (S_n) instantly, in the robot base (B) referential. The central graph of transformations is presented in Figure 7.2 and aims to express the transformation of robot base to each grasping point on the tray.

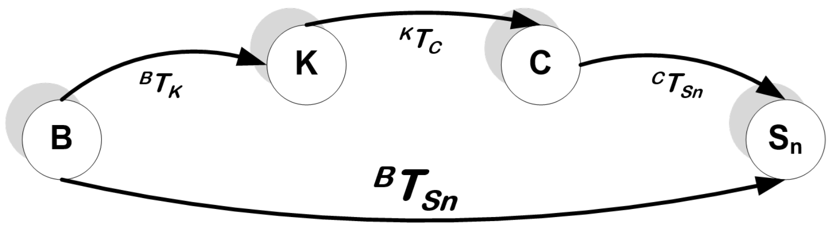


Figure 7.2: Main transformation graph of the robot base (B) to every specimen grasping point (S_n).

The mathematical relation between geometric transformations is defined by Equation 7.1 as

$$ {}^{B}T_{S_n} = {}^{B}T_{K} \times {}^{K}T_{C} \times {}^{C}T_{S_n}, \tag{7.1}$$

where each term's definition and procedure to be obtained are referred below:

- ${}^{B}\mathrm{T}_{S_n}$ - Geometric transformation between the robot base (B) and each specimen grasping point (S_n). It is the final result to achieve and allows the manipulator to receive these correct specimen locations to pick. The result is returned by $Charucodetection$ algorithm;
- ${}^{B}\mathrm{T}_{K}$ - Geometric transformation between the robot base (B) and the camera frame (K). Calculated each time any of both frames need to change locations, being obtained through Hand-Eye calibration (Section 7.1.2). Here, an auxiliary transformation graph is generated to obtain this geometric transformation (Equation 7.8);

[1] n corresponds to the number of specimens in the tray (n= 6).
[2] Field-Of-View

- ${}^{K}\mathrm{T}_{C}$ - Geometric transformation between the the camera frame (K) and the tray's reference marker (C). The transformation required is also obtained by the $Charucodetection$ created using OpenCV resources (see Section7.2).
- ${}^{C}\mathrm{T}_{S_n}$ - Geometric transformation between the tray's reference marker (C) and each specimen grasping point (S_n). The distances of each specimen grasping region (central point) to the tray's reference marker are extracted from CAD design (see Appendix E) to a spreadsheet file and imported to the *Charuco detection* algorithm after that.

7.1 Calibrations

The calibration process aims to obtain the geometric transformations between coordinate systems (positions and orientations) required to implement the system. Note that the complete calibration procedures are performed through the ROS framework. In the end, the intrinsic parameters (camera matrix, distortion coefficients, rectification and projection matrices) and the extrinsic values (transformation matrix between robot base and camera frame) obtained are imported to $Charucodetection$ algorithm (see section 7.2).

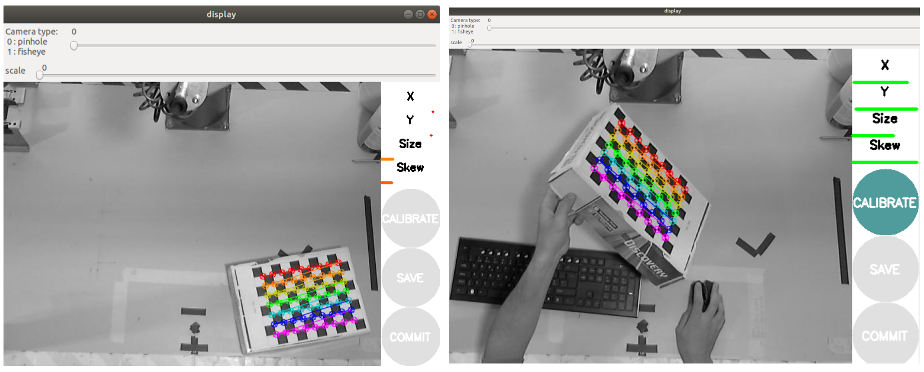
7.1.1 Intrinsic Parameters

The intrinsic parameters represent a projective transformation from the 3D camera's coordinates into the 2D image coordinates. They are responsible for describing the internal parameters of the camera and depend on the focal length, optical centre, skew coefficient and coefficients that describe the lens distortion [68].

The calibration procedure followed in ROS to calibrate the RGB camera (monocular) is the one described in the tutorial [69]. Here, an *OpenCV* calibration algorithm is launched by the *cameracalibrator.py* node (from *camera_calibration* package) and designed to calibrate cameras through a helper GUI (Graphical User Interface). The image raw content is on */usb_cam/image_raw* topic, returned by the node *usb_cam_node* simultaneously launched (from *usb_cam* package). The command lines in Linux terminal which launch the two nodes referred are shown below:

```
rosrun usb_cam usb_cam_node
rosrun camera_calibration cameracalibrator.py --size 9x7 --square 0.025 image:=/usb_cam/
    image_raw camera:=/usb_cam
```

A checkerboard is used as a calibration pattern. The board dimensions are 9×7 squares, with 25 mm size unit. This method is iterative and adjusts the sensor's intrinsic parameters in order to minimize the projection error of each image. Figure 7.3a presents the calibration GUI. The checkerboard is moved, in each iteration, to different positions until all the top right bars reaches green states (see Figure 7.3b). The final results of the intrinsic parameters are presented on Equations 7.2, 7.3, 7.4, and 7.5. These values are extracted to an XML file to further be applied on *Charuco detection* algorithm.



(a) (b)

Figure 7.3: (a) Starting, and (b) final iterations of the calibration for intrinsic parameters.

This intrinsic calibration method is publicly available on `https://www.youtube.com/watch?v=QEqIVUfse30`.

$$\text{camera matrix} = \begin{bmatrix} 852.613017 & 0.000000 & 319.984027 \\ 0.000000 & 855.313321 & 275.982827 \\ 0.000000 & 0.000000 & 1.000000 \end{bmatrix} \tag{7.2}$$

$$\text{distortion} = \begin{bmatrix} 0.086406 & -0.103778 & 0.013355 & -0.001461 & 0.000000 \end{bmatrix} \tag{7.3}$$

$$\text{rectification} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.4}$$

$$\text{projection} = \begin{bmatrix} 868.660767 & 0.000000 & 319.420811 & 0.000000 \\ 0.000000 & 864.929688 & 279.846235 & 0.000000 \\ 0.000000 & 0.000000 & 1.000000 & 0.000000 \end{bmatrix} \tag{7.5}$$

7.1.2 Extrinsic Parameters ($^{B}\mathbf{T}_{K}$)

This section presents the implementations performed to acquire the geometric transformation between the camera's coordinate system (its position and orientation) to the robot base (*base_link*). It is represented by $^{B}T_{K}$ and required to perform when one of the frames changes location. Here, the secondary transformation graph generated to obtain the extrinsic parameters derives from the main transformation graph of the Visual Perception Module already approached (see Equation 7.1).

Mathematically expressing, note that a geometric transformation is a homogeneous matrix which transforms a point to another. Generally, it is represented by

$$T = \left[\begin{array}{ccc|c} & R & & P \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \tag{7.6}$$

where rotation matrix (R) has 3×3 dimension, and it can be determined through the three angles of the referential system that rotate around each one of its axis (x, y, z) to obtain the new transformation (coordinate system). On the other hand, the translation matrix (P), with 3×1 dimension, represents the shifting position from the original reference. Both concepts were reviewed on [70], being expressed by

$$R = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}, \text{ and } P = \begin{bmatrix} px \\ py \\ pz \end{bmatrix}. \tag{7.7}$$

Transformation Graph and Equation

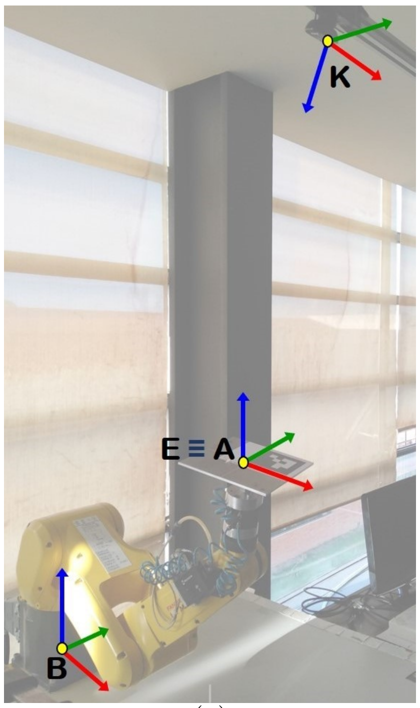
Resulting from the main transformation graph of the system (Figure 7.2), an auxiliary relation is derived to obtain ${}^{B}T_{K}$, being presented on Figure 7.4b. The final result is achieved through a calibration method, multiplying the top three transformations of the graph expressed. This method is known as Hand-Eye Calibration.

The auxiliary graph is mathematically expressed by

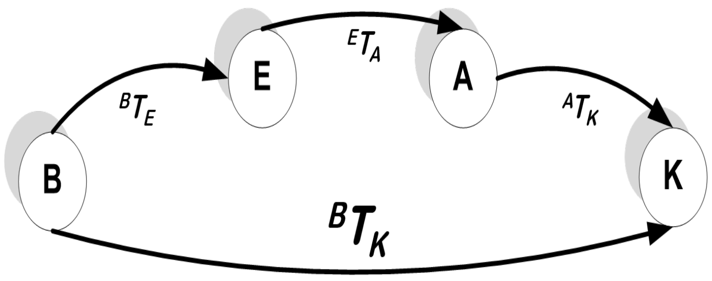
$${}^{B}T_{K} = {}^{B}T_{E} \times {}^{E}T_{A} \times {}^{A}T_{K}, \tag{7.8}$$

being the terms of this transformation described by

- ${}^{B}\mathrm{T}_{K}$ - Geometric transformation between robot base (B) and camera (K) frames. The final result is extracted from the Hand-eye calibration procedure;
- ${}^{B}\mathrm{T}_{E}$ - Geometric transformation between robot base (B) and end-effector (E). Given by the ROS's transformation frames (TF), which return the instant difference between both *base_link* (robot base) and *tool0* (end-effector) frames.
- ${}^{E}\mathrm{T}_{A}$ - Mathematically represented by the identity matrix ($I_{4\times4}$). In order to guarantee that both coordinate systems match in position and orientation ($E = A$), a tool adjustment is performed on the manipulator's end-effector.
- ${}^{A}\mathrm{T}_{K}$ - Geometric transformation between Aruco marker (A) and camera frames (K). The result of this transformation is automatically obtained through the ROS package *moveit_calibration* [71] and it depends on the different Aruco marker positions, while calibration occurs.



(a)



(b)

Figure 7.4: (a) Coordinate systems layout for Hand-Eye Calibration, and (b) transformation graph of this relation .

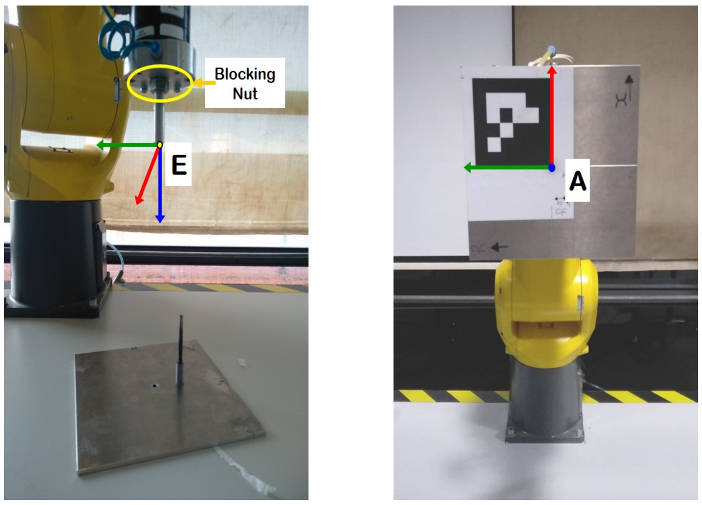
Tool adjustment (${}^{E}\mathbf{T}_{A}$)

This geometric transformation is obtained by matching both coordinate systems (E=A) in position and orientation. By ensuring this similarity, the transformation matrix which represents this relation is equal to an identity matrix ($I_{4\times4}$), i.e., the neutral element in matrices multiplications. The manipulator's suction grip (Figure 7.5a) is replaced by a tool calibrator produced in Simões work [62] and specially designed to the same purpose here. A visual calibration target is placed (see selected Aruco Marker with predefined configurations in Section 5.2.5) in this tool, with its reference (origin, x and y axis) matching with the end-effector's axis (Figure 7.5b).

Then, aiming to match both z axis, the calibrator tool is properly coupled to the manipulator's end-effector. This tool has a screw thread which helps to adjust the z axis depth, and a blocking nut to fix its position (Figure 7.5a). To coincide both TCP axis positions of real and ROS-Industrial manipulator in simulation (on z axis), the ROS Server is initialized with the *fanuc_lrmate200id_moveit_config* package, by executing the command already described in Chapter 5 above:

```
roslaunch fanuc_lrmate200id_moveit_config_moveit_planning_execution.launch sim:=false
    robot_ip:=192.168.0.231
```

The real manipulator is placed on the top of the robot's experimental table, which is



(a) (b)

Figure 7.5: (a) Coordinate system of end-effector (E) and tool calibrator before coupling, and (b) Aruco Marker used in the tool as visual calibration target.

a reference position and represents z equal to 0 in robot base coordinate system (B in real-world and *base_link* in ROS model). The similarity to both TCP z value frames (real and ROS model) is guaranteed by positioning the real end-effector (with the calibrator tool coupled) on the table's surface, which is the reference position that returns z=*0*. Simultaneously, from *tf* package, the ROS node *tf_echo* is launched with the parameters *base_link* and *tool*0 frames, representing the *TF* (Transformation Frames) between robot base and end-effector frames, respectively, in the ROS-Industrial model.

As ROS *rqt_graph* assistant shows in Figure 7.6, the *tf_echo* ROS node (green) subscribes the */tf_static* topic (red) to obtain these frame values published by *robot_state_publisher* node (blue). This subscriber node returns the translation and rotation between both frame.

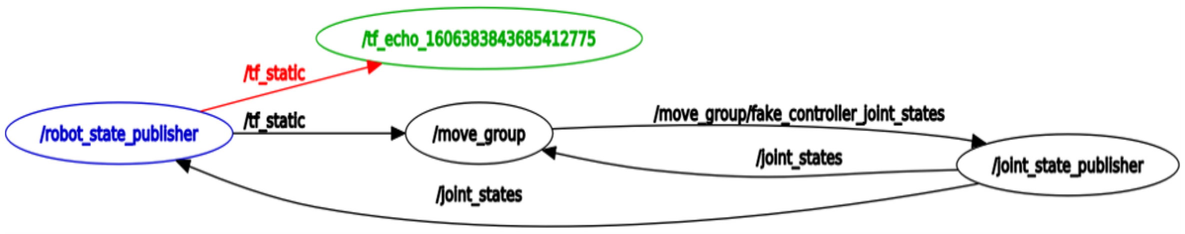


Figure 7.6: ROS *rqt_graph* assistant shows interaction between nodes launched.

Therefore, the z translation value is regulated to 0 (included on Table 7.1 data) by moving the real manipulator on teach pendant, while a refined adjustment is performed on the screwed tool calibrator, until it perfectly contacts with the table's surface (Figure 7.7). After that, the locking nut is used to fix its position, and the identity matrix of ${}^{E}T_{A}$ is guaranteed because both TCP positions, real and simulated models, have z=0.

Table 7.1: ROS node *tf_echo* node final outcome obtained in Linux command line (terminal).

Translation (x, y, z)		0.443	-0.011	**0.000**	-
Rotation	Quaternion	0.729	-0.684	-0.002	0.003
	RPY (radian)	3.135	-0.002	-1.508	-
	RPY (degree)	179.630	-0.097	-86.376	-

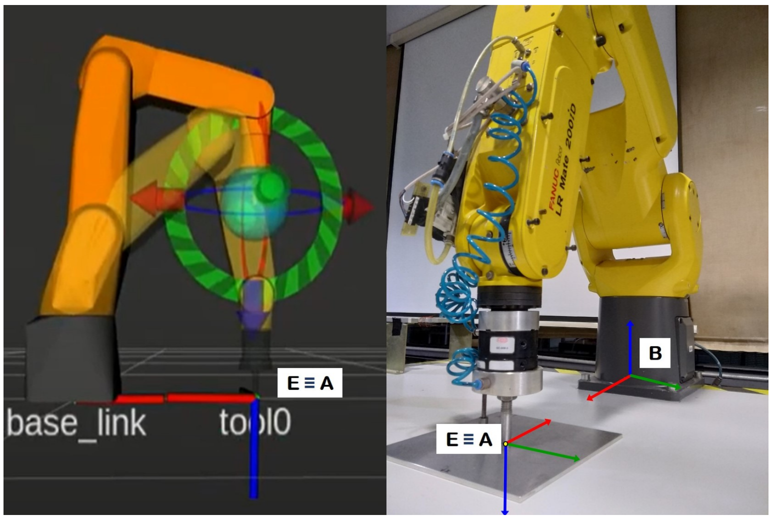


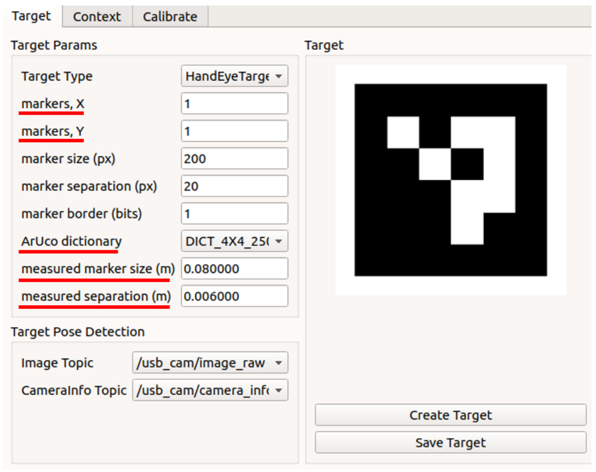
Figure 7.7: Final tool adjustment of z coordinate to match TCP positions of real and ROS-Industrial manipulators.

Hand-Eye Calibration

This calibration is performed using an assistant plug-in GUI of *MoveIt!*. At this point, all the elements required are set to get the $^{B}\mathrm{T}_{K}$ geometric transformation. This parameter is obtained through Hand-Eye calibration method [72], using the new hand-eye tool from the *moveit_calibration package* [71], which is plug-in based. This GUI provides:

- RVIZ plug-in - Set the calibration parameters, make the calibration control and call the other two plug-ins;
- Default target plug-in - Create the target image and detect target pose;
- Default Solver plug-in - Solve the camera-robot pose throughout samples collected of the manipulator's different positions to convert to $^{B}\mathrm{T}_{K}$ later, based on the transformation graph derived (see Figure 7.4).

The pre-requirements executed before launching this interface are (i) establish the ROS Server connection between PC and manipulator's controller (using the command line already referred above from *fanuc_lrmate200id_moveit_config* package), and (b) launch the camera node publishing images on */usb_cam/image_raw* topic.



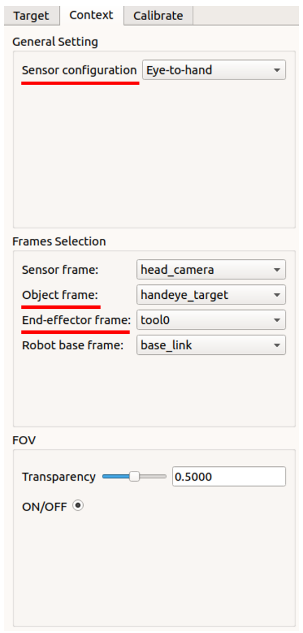
Target Params

Target Type	HandEyeTarge
markers, X	1
markers, Y	1
marker size (px)	200
marker separation (px)	20
marker border (bits)	1
ArUco dictionary	DICT_4X4_25(
measured marker size (m)	0.080000
measured separation (m)	0.006000

Target Pose Detection

Image Topic	/usb_cam/image_raw
CameraInfo Topic	/usb_cam/camera_inf(

(a)



General Setting

Sensor configuration	Eye-to-hand

Frames Selection

Sensor frame:	head_camera
Object frame:	handeye_target
End-effector frame:	tool0
Robot base frame:	base_link

FOV

Transparency	0.5000
ON/OFF	●

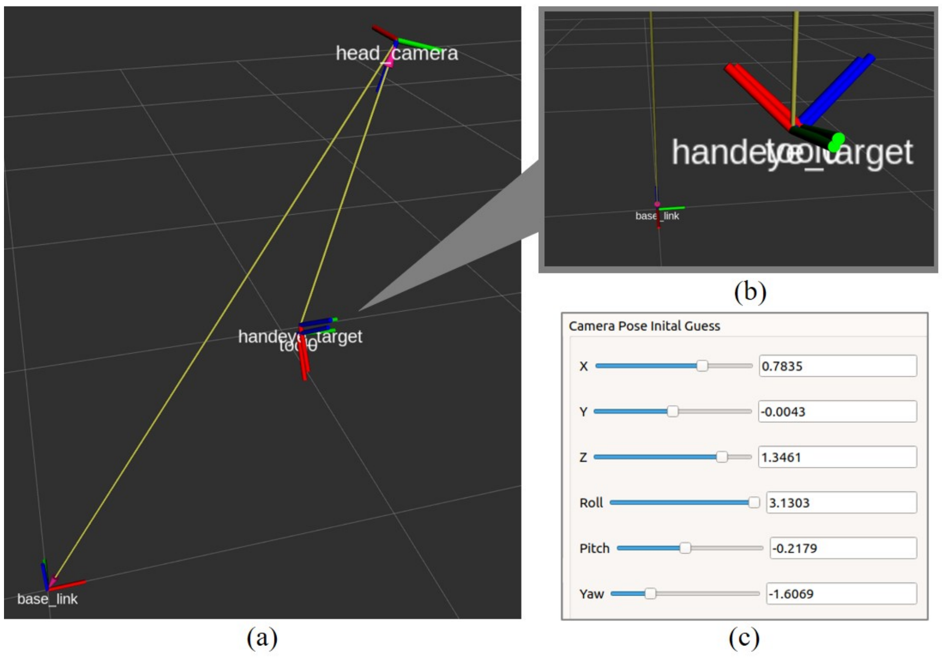
(b)

Figure 7.8: (a) Target, and (b) Context tabs from *MoveIt!* GUI for calibration.

Intructions from tutorial [73] are followed in order to perform the initial configurations. On the first Target tab (Figure 7.8a), the Aruco parameters are selected according to the marker's features used for this work (see Section 5.2.5) and assembled to the calibrator tool (Figure 7.5b). For target detection purposes, the info and image topics (which *usb_cam_node* launched subscribes) are selected (*/usb_cam/camera_info* and */usb_cam/image_raw* topics, respectively).

The sensor configuration selected on the Context tab (Figure 7.8b) is the *Eye-to-hand* type, which represents the type of Hand-Eye calibration where the target/Aruco marker is attached to the end-effector ($E{=}A$), and the camera frame is fixed. Note that while calibration occurs, the procedure shall collect the sufficient number of samples capable of performing the best match between both Aruco and End-effector frames (*handeye_target* and *tool0*). See visual representations of Figures 7.9a and 7.9b.

On the third Calibrate tab (see Figure 7.10), the solver selected from the loaded plug-in is based on [74] optical tracking system. At this point, all the parameters are set to enable the interactive calibration. Here, a collection of samples are captured while the robot goes through different points. In each step, the correlation between *base_link-to-tool0* and *head_camera-to-handeye_target* poses improves the similarity between *tool0* and *handeye_target* frames. The number of samples collected for this calibration is 15 (minimum recommended on [73]).



Camera Pose Inital Guess

X	0.7835
Y	-0.0043
Z	1.3461
Roll	3.1303
Pitch	-0.2179
Yaw	-1.6069

Figure 7.9: (a) Pose estimation of the 13th iteration of the calibration (with 13 samples collected), (b) zoom fit of the matching frames tracked, and (c) final camera pose estimated.

Figure 7.10 presents the first poses sample captured. Note that *TF* frames highlighted (with message Pose format[3]) correspond to the referred geometric transformations (${}^{B}\mathrm{T}_{E}$ and ${}^{A}\mathrm{T}_{K}$) if converted. The final result presented in Figure 7.9c, camera's pose guess, describes the estimated position of the camera frame to the robot base in the real-world.

In this ROS method, the format of camera *Pose* data returned is (i) translation between frames given by x, y and, z coordinates and, (ii) orientation in Euler angles (Roll, Pitch and Yaw). Thereafter, this result is exported to the *CharucoDetection* algorithm (see next Section 7.2), where the conversion of this result obtained to the final ${}^{B}\mathrm{T}_{K}$ geometric transformation format is performed.

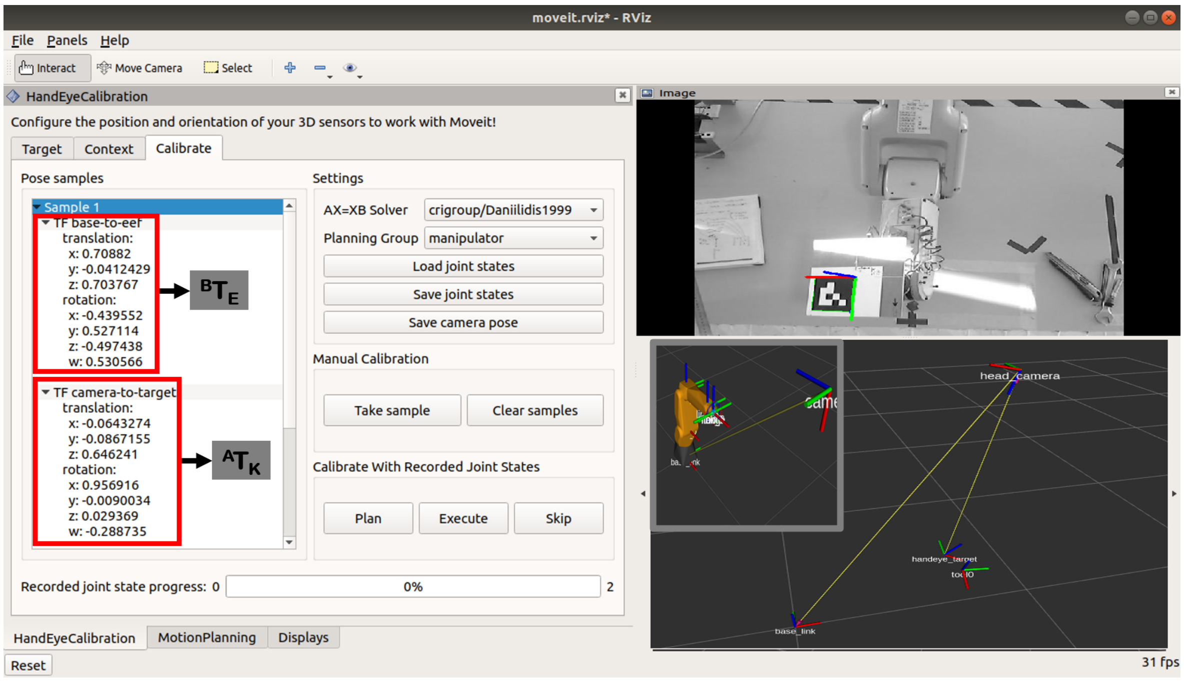
Demonstration of the complete extrinsic calibration procedure (tool adjustment + Hand-Eye calibration) is available on video in `https://www.youtube.com/watch?v=0MHRzPrFhzg`.

7.2 Charuco Detection algorithm(${}^{B}\mathbf{T}_{S_n}$)

The intrinsic and extrinsic calibrations are set before the system tracker starts operating (in offline mode):

- The intrinsic parameters are only required to be calculated on the first use of the camera;

[3]`http://docs.ros.org/en/jade/api/geometry_msgs/html/msg/Pose.html`



Configure the position and orientation of your 3D sensors to work with Moveit!

Pose samples

Sample 1
TF base-to-eef
translation:
x: 0.70882
y: -0.0412429
z: 0.703767
rotation:
x: -0.439552
y: 0.527114
z: -0.497438
w: 0.530566
TF camera-to-target
translation:
x: -0.0643274
y: -0.0867155
z: 0.646241
rotation:
x: 0.956916
y: -0.0090034
z: 0.029369
w: -0.288735

Settings

AX=XB Solver: crigroup/Daniilidis1999
Planning Group: manipulator

Manual Calibration

Calibrate With Recorded Joint States

Recorded joint state progress: 0 0% 2

Figure 7.10: Calibrate tab where the interactive Hand-Eye calibration is performed.

- The extrinsic calibration must only be performed if the robot or camera hardware are moved.

On the other hand, the Charuco detection algorithm runs alongside with the system operation (online mode) because each time the system is started, a new capture of tray position is required to get the new specimen grasping points. The code is built in C++ language in Visual Studio using OpenCV library. Its main goal is to return the coordinates to each specimen of the tray (with random position), or also known as $^{B}TS_n$ geometric transformations, to further send to the manipulator's controller in coordinates and Euler angles format. When the program is imported, the set of files already collected on this work are loaded. As represented on Figure 7.11, the three different features of the code (grey in the flowchart) run simultaneously to get to the final result, representing separated routines to obtain the transformations required (Figure 7.2).

The input data for this code (red on the flowchart) are (i) the predefined parameters for the Charuco board (see Section 5.2.5), and (ii) the self-defined rotation matrix to the tray's referential, corresponding to the orientation which end-effector picks each specimen (represented in Figure 7.12). Here, the rotation matrix defined is

$$^{C}R_{Sn} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}. \tag{7.9}$$

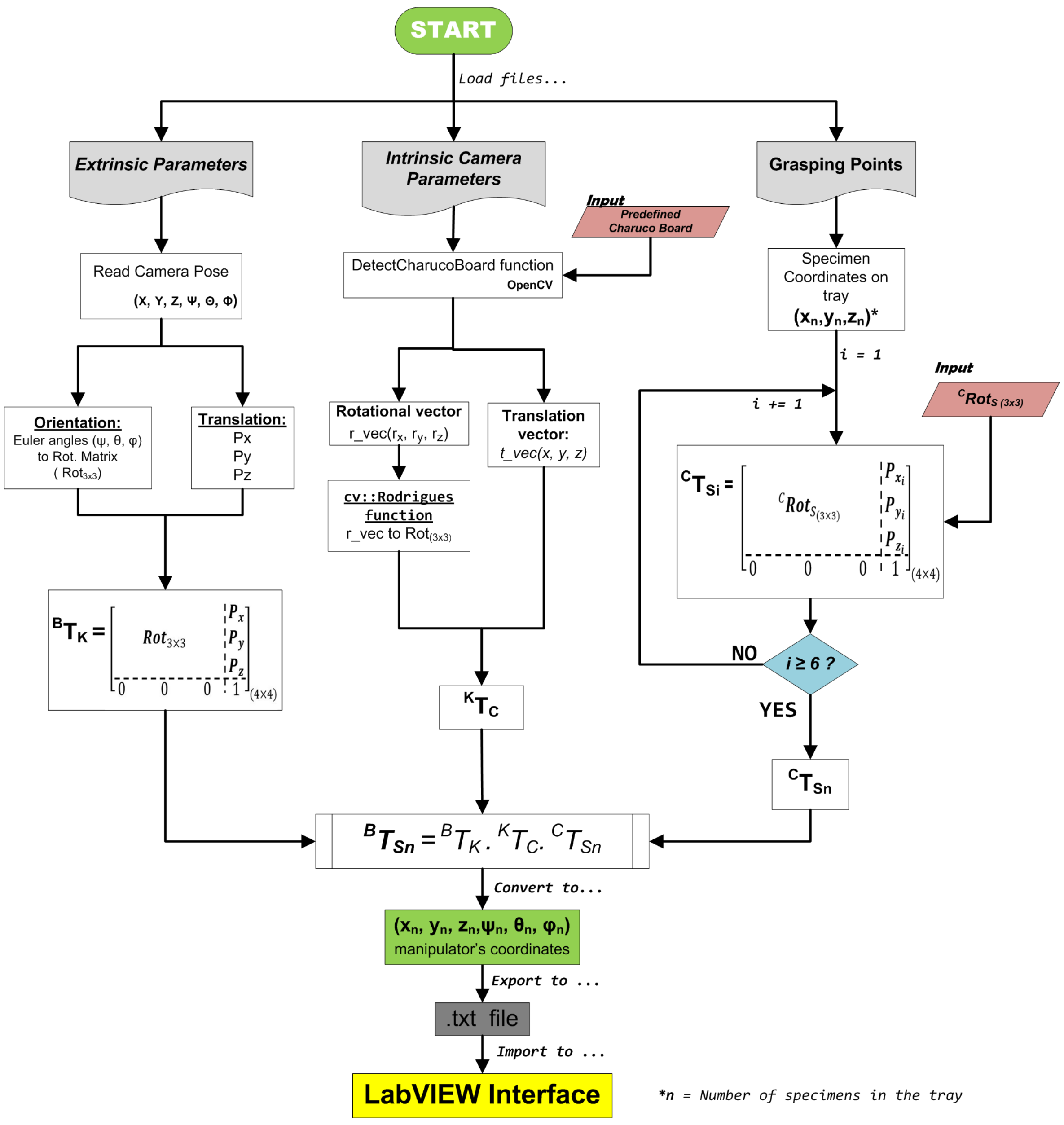

START
Load files...
Extrinsic Parameters
Intrinsic Camera Parameters
Grasping Points
Input
Predefined Charuco Board
Read Camera Pose
(X, Y, Z, Ψ, Θ, Φ)
DetectCharucoBoard function
OpenCV
Specimen Coordinates on tray
(x_n,y_n,z_n)*
i = 1
Input
$^{C}Rot_{S\,(3x3)}$
i += 1
Orientation:
Euler angles (ψ, θ, φ) to Rot. Matrix (Rot_{3x3})
Translation:
Px
Py
Pz
Rotational vector
r_vec(r_x, r_y, r_z)
Translation vector:
t_vec(x, y, z)
cv::Rodrigues function
r_vec to $Rot_{(3x3)}$
$^{C}T_{Si} = \left[\begin{array}{ccc:c} & ^{C}Rot_{S(3x3)} & & P_{x_i} \\ & & & P_{y_i} \\ & & & P_{z_i} \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right]_{(4x4)}$
$^{B}T_{K} = \left[\begin{array}{ccc:c} & Rot_{3x3} & & P_x \\ & & & P_y \\ & & & P_z \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right]_{(4x4)}$
NO
$i \geq 6$?
YES
$^{K}T_{C}$
$^{C}T_{Sn}$
$^{B}T_{Sn} = {}^{B}T_{K} \cdot {}^{K}T_{C} \cdot {}^{C}T_{Sn}$
Convert to...
$(x_n, y_n, z_n, \psi_n, \theta_n, \varphi_n)$
manipulator's coordinates
Export to ...
.txt file
Import to ...
LabVIEW Interface
*n = Number of specimens in the tray

Figure 7.11: Main flowchart of Charuco's Detection algorithm.

The extrinsic routine, in the main code, imports the camera pose results from the Hand-Eye calibration (see Section 7.1.1). The orientation part of this data (Euler angles) is converted to a rotation matrix, with 3×3 dimension, and is merged with the translation points (Px, Py, Pz) from the same initial source, resulting in the homogeneous geometric transformation needed $^{B}T_{K}$.

The intrinsic camera parameters routine has embedded the essential tool for this code: the Charuco detector. Initially, the intrinsic parameters (see Section 7.1.2) are loaded and assigned to the camera used on *detectCharucoBoardWithCalibrationPose()* function [75] for

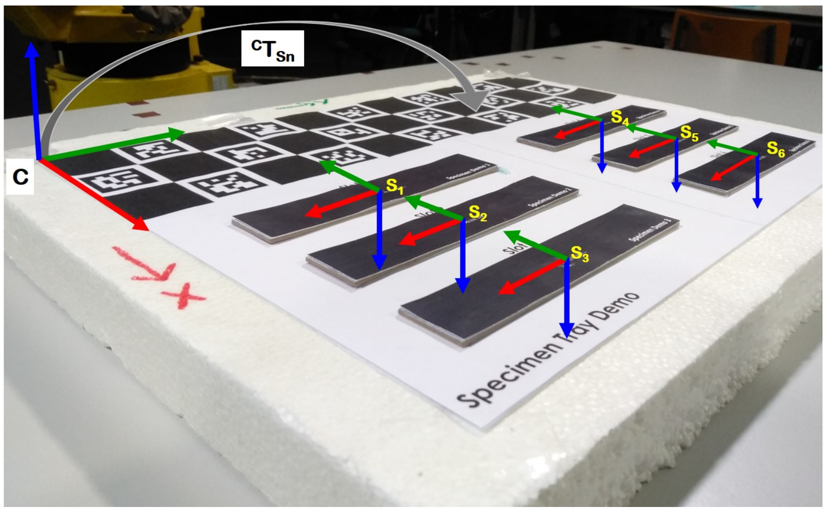


Figure 7.12: Relation between Charuco and specimen frames to pick.

detection of Charuco corners. The results returned by this function are (i) the rotational vector *r_vec*, and (ii) the translation vector *t_vec*. The *r_vec* vector is consequently converted to a Rotation matrix, and further merged on the final geometric transformation ${}^{K}T_{C}$. This conversion (*r_vec* to $Rot_{3\times3}$) is performed by Rodrigues function[4].

For the Grasping Points routine, the coordinates are provided through a text file (detailed on Section E) and they refer to the distance of their grasping regions (position and orientation) to the tray's origin. Note that this referential is self-defined and included as an input of the system (see Figure 7.12). Next, a loop of six indexes is performed, merging the six specimen grasping points with the rotational matrix, resulting in a vector of vectors of ${}^{C}T_{Sn}$ (in a homogeneous geometric transformation format).

At this point, all the transformation results are multiplied, resulting in the final ${}^{B}T_{Sn}$ geometric transformation. Here, a function in OpenCV is applied to convert the rotation matrix part of the transformation - ${}^{B}R_{Sn\ (3\times3)}$ to Euler angles ($\Psi_n, \theta_n,$ and φ_n[5]) in degrees, and the translation matrix - ${}^{B}P_{Sn\ (3\times1)}$ to coordinates (x_n, y_n and z_n) format. Function created is based on [77] examples. The final result returned are $(x_n, y_n, z_n, \Psi_n, \theta_n, \varphi_n)$ coordinates in the manipulator's valid format.

As the last step, the results are exported to a text file ready to be further imported to the LabVIEW interface.

[4]**cv::Rodrigues**- Converts a rotation matrix to a rotation vector or vice versa [76].
[5]yaw, pitch, and roll

Chapter 8

Robotic Module

The strategy used to perform the specimen handling procedure is reported in this chapter. Here, the FANUC LR Mate 200iD robot controller shares a TCP/IP communication protocol with the PC through an Ethernet cable connection. The interface with the LabVIEW to control the manipulator is guaranteed with the DigiMetrix Robotics Library available for FANUC [25]. The library setup and controller configurations are described in the next sections, as well as the final block diagram with the code required to perform this routine.

8.1 Library setup and controller configuration

The first step is to install the Fanuc Robotics library provided by DigiMetrix in LabVIEW through its VI[1] Package Manager. The library server programs for robot controller (available on the directory of Digimetrix package installed [25]) are uploaded. The list of files is selected depending on the R-30iB controller software version (version V8.20), and is located on the installed directory path: `/DigiMetrix/FanucRoboticsLibrary/Serverbinaries/R-30iB/V8.20/`. The list of programs, within this folder, which need to be uploaded to the controller are:

- *dmbuf.pc*;
- *dmcfg.pc*;
- *dmcmd.pc*;
- *dmlog.pc*;
- *dmmove.ls*;
- *dmpkt.pc*;
- *dmserver.pc*;
- *dmsocket.pc.*

The method used to upload all programs on the robot controller is FTP client program (FileZilla[2] software on computer), which supports DOS mode as Server type. The FTP server on robot controller is setup by default, and enables the file transfer. In manipulator's teach pendant, the *DMSERVER.PC* file is the main program, which runs all other files. Before its first start, in teach pendant $[SYSTEM] \rightarrow [VARIABLES]$, the $\$KAREL\_ENB$ variable is set with value 1, in order to in order to allow Karel programs (the ones with *.PC extensions)

[1]Virtual Instrument
[2]https://filezilla-project.org

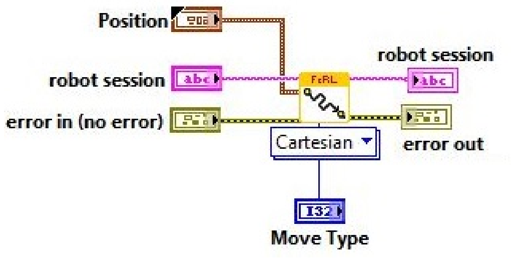
to show up in teach pendant display. At the first start, this main program enables the automatic configuration of controller variables needed for communication and the automatic start-up at the controller boot. The controller settings automatically changed by the Server program are presented next:

- $SHELL_CFG.$SHELL_EXT = FALSE
- $SHELL_CFG.$SHELL_NAME = 'DMSERVER'
- $SHELL_WRK.$KAREL_SOP = TRUE
- $HOSTS_CFG[4].$COMMENT = 'DM_FcRL_Svr'
- $HOSTS_CFG[4].$PROTOCOL = 'SM'
- $HOSTS_CFG[4].$SERVER_PORT = 10000
- $HOSTS_CFG[4].$OPER = 3
- $HOSTS_CFG[4].$TIMEOUT = 9999

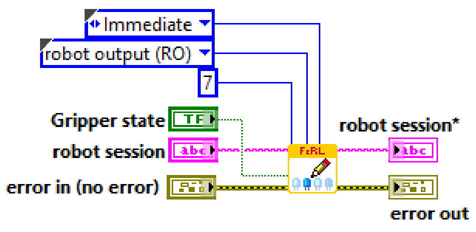
These settings install *DMSERVER* program as a shell program and configure server Tag 4 (S4) to be used as Socket Messaging server on port 10000. After that, the Server is started automatically at each controller's boot, and the manipulator is ready to execute the LabVIEW program, using Digimetrix Robotics library blocks for Fanuc, to perform the specimen picking intended on the tray.

8.2 LabVIEW Pick and Place routine

In LabVIEW, the control of the manipulator is performed through code blocks. Here, the pick and place code executes the number of times equal to the number of specimens in the tray (n=6). The essential blocks on the algorithm are the *MoveSynchro* (see Figure 8.1a) and *SetDigitalLine* (see Figure 8.1b). The *MoveSynchro* block moves the arm using point-to-point motion (synchro motion), from the current position to the specified point received as input (with Cartesian coordinates format). Note that this input data type is provided thought a type definition (*typedef*) default structure defined by the Digimetrix Robotics library for Fanuc (see Figure 8.2).



(a)



(b)

Figure 8.1: (a) *MoveSynchro*, and (b) *SetDigitalLine* blocks from Fanuc Robotics library.

The resulting picking coordinates from *Charuco detection* algorithm (x_n, y_n, z_n, w_n, p_n, r_n) are appended in their proper fields of this structure, using a LabVIEW method created to enable the correct coordinates format as input on the moving block (see this method on Appendix G). Additionally, the *move type* defined for this motion block is *linear*.

The other diagram block referred above (*SetDigitalLine*) sets the digital output line state on the manipulator, also defined as Robot Output (RO) register. The *Gripper state* input controls the digital pressure switch (SMC ZK2 vacuum ejector[3]), enabling the pneumatic/suction grip (coupled to the manipulator's end-effector) action. The digital line which controls the pressure switch action is *RO=7*, and its *Gripper state* input depends on the manipulator's action (False and True for gripping and placing specimens, respectively). This function is executed immediately by defining its execution mode input of this block (*exec_mode* = Immediate).

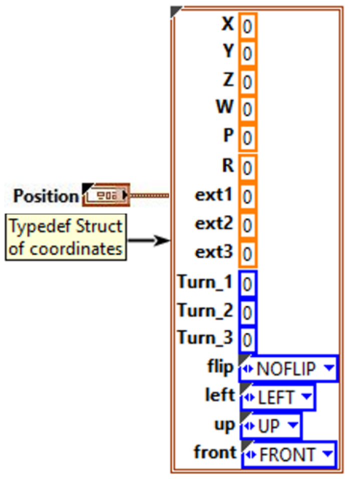


Figure 8.2: Type definition structure of the data input block coordinates for manipulator.

The subVI[4] modules are embedded parts of the code similar to a sub-routine and represented in the main VI (Virtual Instrument Environment) by an icon. A modular code for manipulator motions (pick or place specimens) is built, and its detailed block diagram is presented in Figure 8.3. Here, the pick and place tasks depend on the *Position* and the *Gripper state* inputs received. Moreover, the approach position blocks are also included, defining 50 mm distance (in the positive z-axis direction) from the destination point. These are defined as arrival and departure positions to reach before and after the destination point, respectively.

Therefore, the main VI, including the complete manipulator motions in a loop, is expressed in Figure 8.4. The procedure always starts and finishes with a "home position"

[3]`https://www.smcpneumatics.com/ZK2-ZSVB-A.html`
[4]LabVIEW sub Virtual Instrument

instruction for the robot's joint states. Within the loop, the *motionSubVI* blocks represent the pick and place specimens routine. Each block receives different coordinates and *Gripper states.* The Cartesian coordinates received in specimen Pick (*MotionSubVI* block) task is provided by the subVI designed to convert the specimen grasping points to the *typedef* structure data received by motion blocks in this library (see Appendix G for this subVI). In the specimen Place task, the input coordinates for *MotionSubVI* block are already in this format, and they are provided by a manual taught position using the teach pendant.

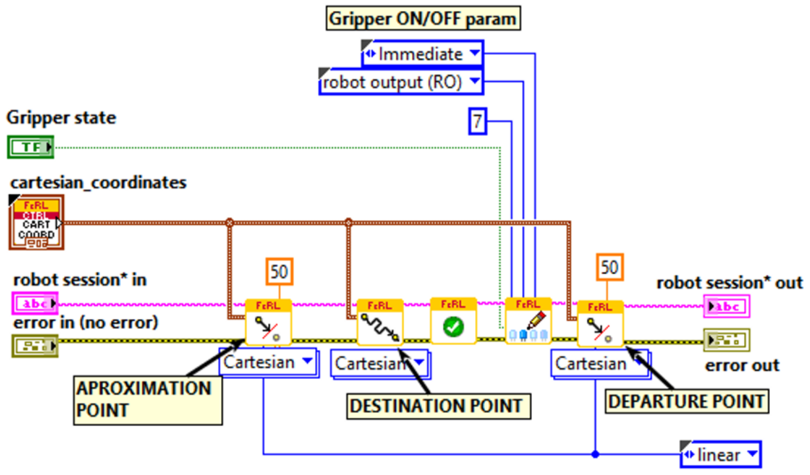


Figure 8.3: LabVIEW subVI routine for pick or place motion of the manipulator - *MotionSubVI.*

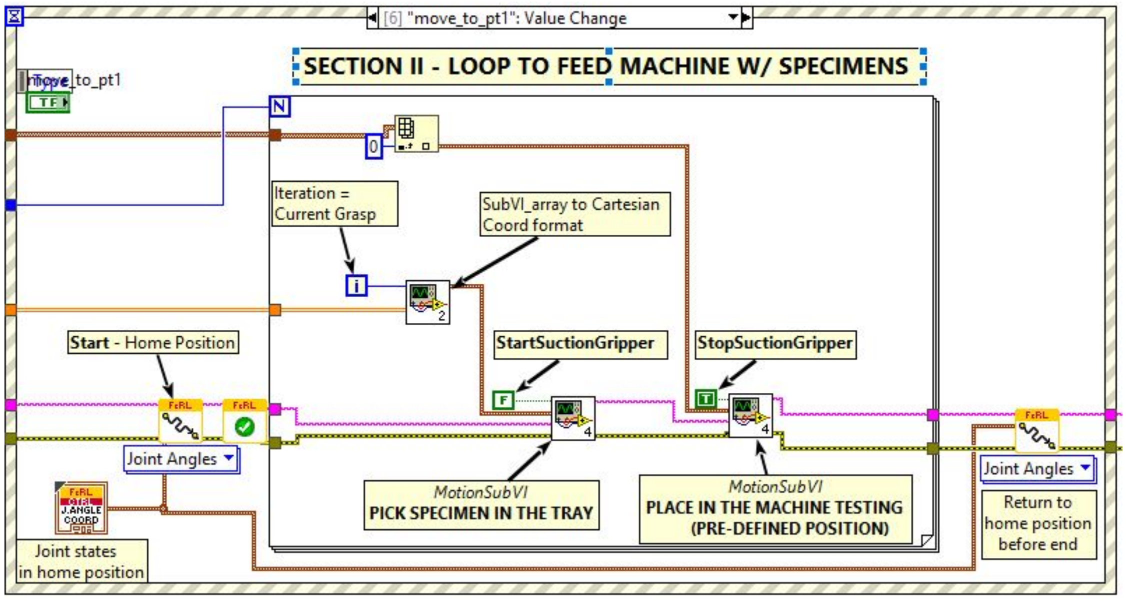

[6] "move_to_pt1": Value Change
move_to_pt1
SECTION II - LOOP TO FEED MACHINE W/ SPECIMENS
Iteration = Current Grasp
SubVI_array to Cartesian Coord format
StartSuctionGripper
StopSuctionGripper
Start - Home Position
Joint Angles
MotionSubVI PICK SPECIMEN IN THE TRAY
MotionSubVI PLACE IN THE MACHINE TESTING (PRE-DEFINED POSITION)
Joint Angles
Return to home position before end
Joint states in home position

Figure 8.4: Main LabVIEW VI of the manipulator motions.

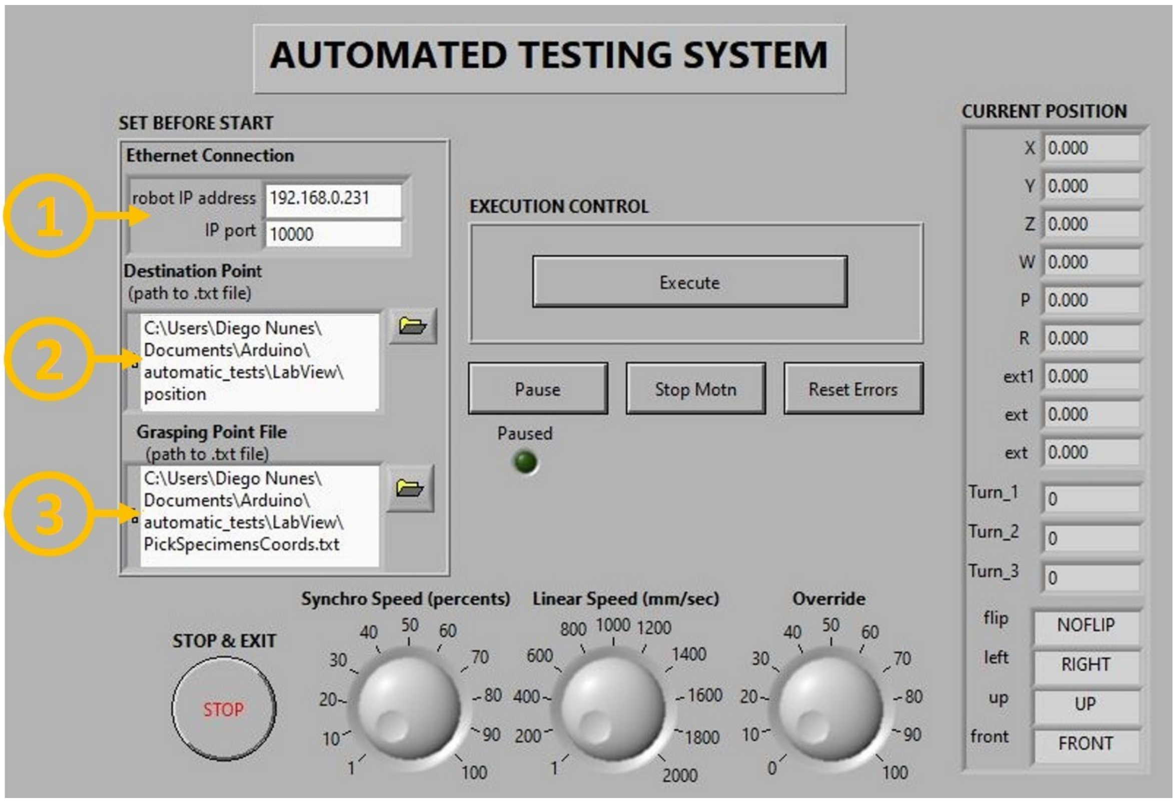
Part VI

Results

Chapter 9

System Integration

This chapter presents the settings required to integrate the different modules developed and consequently enabling the system operation. Figure 9.1 presents the main user interface built. As pointed out, three configurations must be set before start: (1) Ethernet connection, (2) destination point, and (3) grasping points file import. After all these pre-selections, the execution command (*Execute*) operates the pick (on a tray with a random position) and place (on a pre-determined position taught before) of each specimen.



SET BEFORE START

Ethernet Connection

robot IP address 192.168.0.231

IP port 10000

Destination Point (path to .txt file)

C:\Users\Diego Nunes\Documents\Arduino\automatic_tests\LabView\position

Grasping Point File (path to .txt file)

C:\Users\Diego Nunes\Documents\Arduino\automatic_tests\LabView\PickSpecimensCoords.txt

CURRENT POSITION

X	0.000
Y	0.000
Z	0.000
W	0.000
P	0.000
R	0.000
ext1	0.000
ext	0.000
ext	0.000
Turn_1	0
Turn_2	0
Turn_3	0
flip	NOFLIP
left	RIGHT
up	UP
front	FRONT

Figure 9.1: Main user interface for automated testing system.

The robot session is initialized by the *IP port* 10000 and *IP address* 192.168.0.231 . Figure 9.2 presents the block diagram which enables this connection. Note that, for trials/security purposes, a *GetSpeed* block is added to the code in order to control the *linear* or *synchro* velocities of the manipulator by knob while testing occurs.

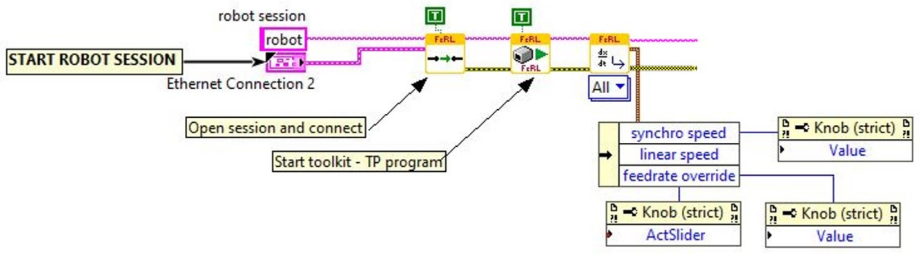


Figure 9.2: LabVIEW block diagram for the robot open session.

9.1 Destination point (Place)

The destination point, considered as the simulated feed position in the testing machine, is taught beforehand to the manipulator using the teach pendant. The pre-defined *FcRL-ex6-saving_positions_to_xml* LabVIEW program [25], provided by the Fanuc Robotics library examples, is the method used to teach and export the destination point to a XML file (in Cartesian coordinates format). Then, before starting the system in the main interface, the file with the destination coordinates is uploaded (Number 2 on Figure 9.1 interface) thought a *read file function* block. This import procedure includes a conversion to *typedef* structure afterwards, with the Cartesian coordinates format required. The Figure 9.3 represents this method on the LabVIEW block diagram created.

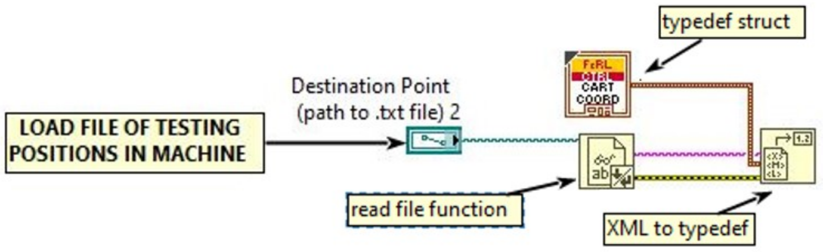


Figure 9.3: LabVIEW block diagram to import XML file of the position taught.

9.2 Grasping points (Pick)

The *Charuco detection* algorithm exports the grasping points to a spreadsheet format file (see an illustrated example of this sort of data organization on Table 10.1 below). In the file to be read, the delimiter between data points is the space key character. Before starting the system, this text file is imported (Number 3 on Figure 9.1) by the *read file function* block, and the conversion of the spreadsheet read to an array is performed. Here, the data type (float-*%f*) and delimiter (space key character) are configured on its function block. After returning the entire array, a sub-routine is simultaneously performed to obtain the array size (lines) which corresponds to the number of specimens in the tray. The result is defined as the count terminal number on the loop created (which executes this number of times the code within). Moreover, it is consequently used as input on the pick and place routine ahead on the main code.

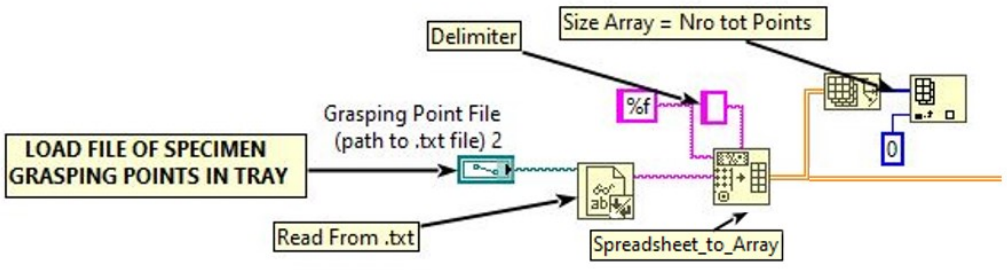

Delimiter
Size Array = Nro tot Points
%f
0
Grasping Point File (path to .txt file) 2
LOAD FILE OF SPECIMEN GRASPING POINTS IN TRAY
Read From .txt
Spreadsheet_to_Array

Figure 9.4: LabVIEW block diagram to upload grasping points from a text file.

All these configuration modules are then integrated to the robotic module routine already detailed on previous Chapter 8, resulting in the main code represented LabVIEW block diagram shown in Appendix H.

Chapter 10

Experiments

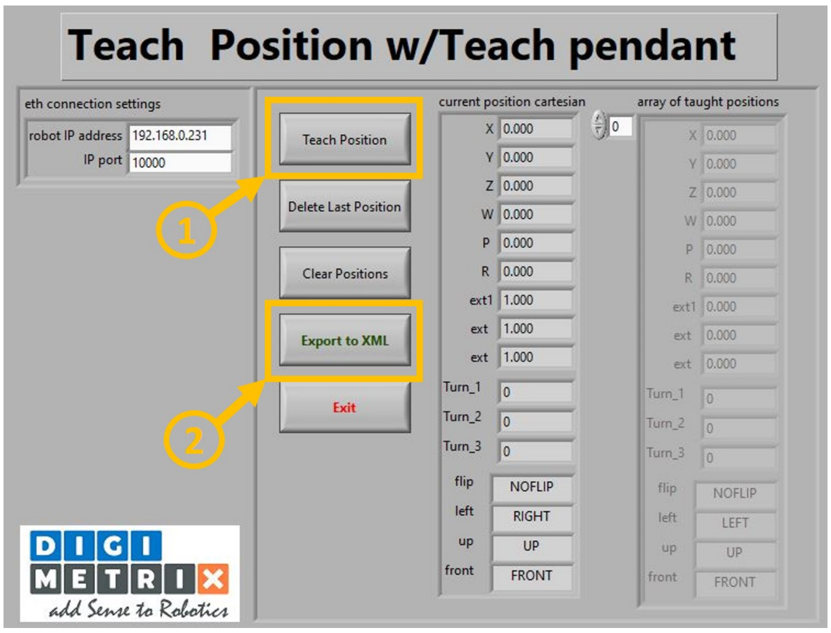
This chapter presents several tests performed to validate in a simulated environment the developed communication structure of the automated testing system, and the performance of the designed Vision Module features.

10.1 Demonstrations

For this particular demonstration case, two tests were performed choosing two random positions to place the tray (reachable to the manipulator's arm) and using the suction grip coupled to the end-effector to handling the specimens. The order of the operation follows the next steps: (1) teach position, (2) detection of the tray reference, (3) set connection and load coordinate files, and (4) execute Pick&Place (specimen handling) system. The first three steps are set before the system starts/execution command.

Teach Feeding position

This task is manually performed using manipulator's teach pendant. It aims to teach a position, simulating the feeding point on a testing machine and exports the results to an XML file. The *FcRL-ex6-saving_positions_to_xml* program is used to perform this task. Its interface is presented in Figure 10.1.



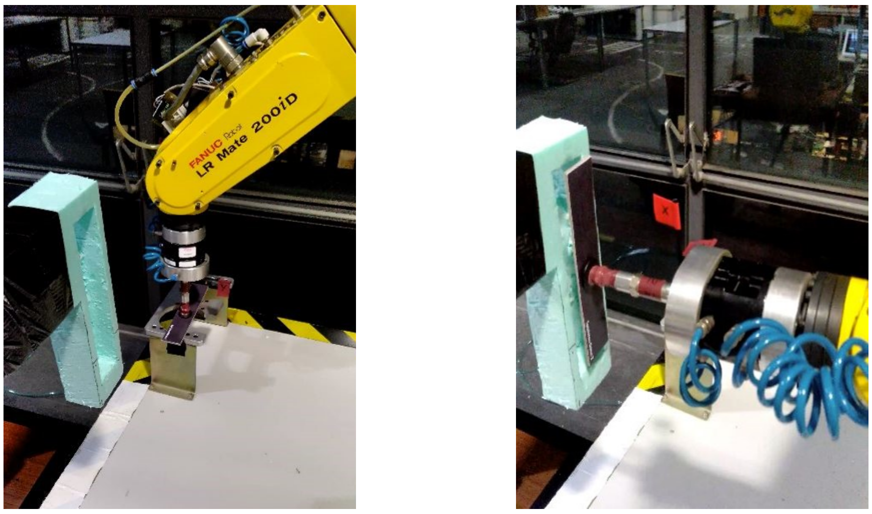
eth connection settings

robot IP address	192.168.0.231
IP port	10000

	current position cartesian	array of taught positions (0)
X	0.000	0.000
Y	0.000	0.000
Z	0.000	0.000
W	0.000	0.000
P	0.000	0.000
R	0.000	0.000
ext1	1.000	0.000
ext	1.000	0.000
ext	1.000	0.000
Turn_1	0	0
Turn_2	0	0
Turn_3	0	0
flip	NOFLIP	NOFLIP
left	RIGHT	LEFT
up	UP	UP
front	FRONT	FRONT

Figure 10.1: *FcRL-ex6-saving_positions_to_xml* LabVIEW program to teach positions [25].

First, the manipulator's arm, with its suction grip grasping a specimen, is taken to the destination required. As examples, the positions from Figure 10.2a and Figure 10.2b were chosen. Then, in the interface to teach positions, the *Teach position* command is pressed to save both. Finally, the *Export to XML* command must be executed to export data to a file.



(a) (b)

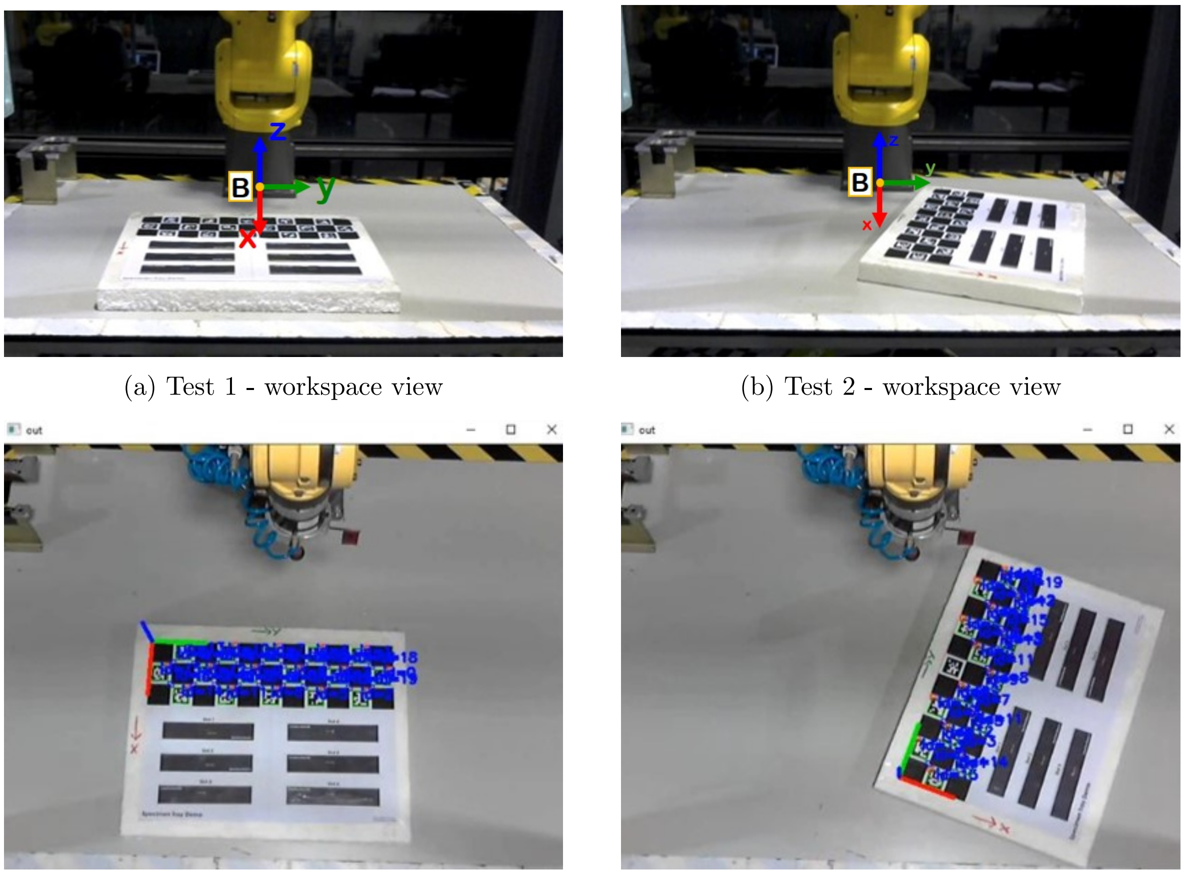
Figure 10.2: Taught Positions (a) 1 and (b) 2.

Detect Tray Reference

The specimens tray was placed into two different/random positions in the workspace. Positions are reachable to the manipulator's arm and within the camera's FOV. See position selected for Test 1 in Figure 10.3a, and for Test 2 in Figure 10.3b. Then, the *Charuco detection* algorithm is executed. As seen in Figures 10.3c and 10.3d, the algorithm displays the detected tray references. When the *Enter* key is pressed on the keyboard, the code returns the Cartesian coordinates calculated of the several specimen grasping point distances, within the tray, to the robot base. Table 10.1 presents the coordinates returned of the specimen grasping regions on the tray on Test 1 position. These results are exported to a text file, formatted with a space delimiter between data points.

Table 10.1: Results of the specimen grasping regions (coordinates) for Test 1 tray's position.

		x (mm)	**y (mm)**	**z(mm)**	**Ψ (º)**	**θ (º)**	**φ(º)**
Specimen Index	**0**	540.054	-153.589	27.1943	-179.46	-0.196987	90.402
	1	595.051	-153.201	26.6757	-179.46	-0.196987	90.402
	2	650.047	-152.814	26.1571	-179.46	-0.196987	90.402
	3	538.511	66.4044	27.9506	-179.46	-0.196987	90.402
	4	593.507	66.792	27.432	-179.46	-0.196987	90.402
	5	648.503	67.1797	26.9134	-179.46	-0.196987	90.402



(a) Test 1 - workspace view

(b) Test 2 - workspace view

(c) Test 1 - Detection in VS (camera view)

(d) Test 2 - Detection in VS (camera view)

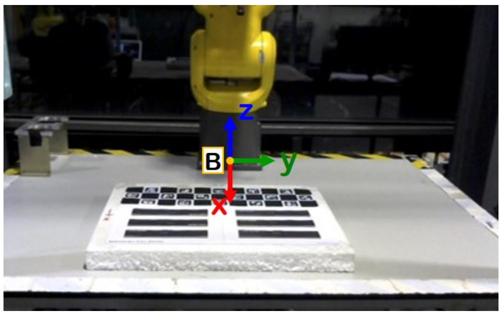
Figure 10.3: Tray's position for Test 1 (in the first column), and for Test 2 (the second column).

Set connection and load coordinate files

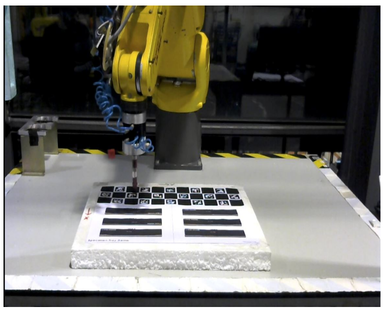
As detailed in Figure 9.1 from Section 9, the required three configurations are set before the system starts: (1) Set IP address and port, (2) Load XML file saved with feeding/taught positions, and (3) load grasping point text file. Note that, for demo purposes, the only taught position used to place the specimen in these tests is the Position 1 (Figure 10.2a).

Execute command

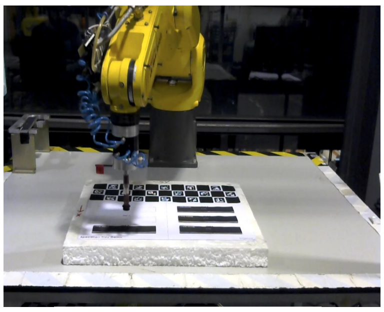
After all the pre-configurations set before the start, the Pick and Place operation is executed by pressing the *Execute* command in the created main LabVIEW interface (see Figure 9.1 interface). Note that the first action of the manipulator is always to move its joints to the home position defined, regardless of its current position (Figure 10.4a). Furthermore, the last action of the manipulator (after performing the complete routine) is to return to the home position as well. Illustrative results of Test 1 are shown in Figures 10.4. Here, Figures 10.4b, 10.4c, and 10.4d represent the approach points of the gripper to the specimens, and Figures 10.4e, 10.4f and, 10.4g each grasping point of the gripper.



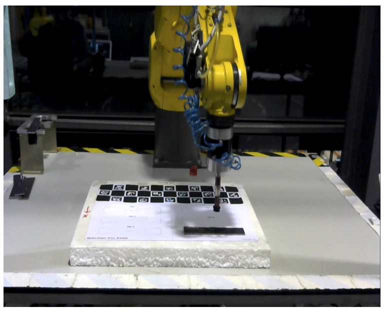
(a) Home Position



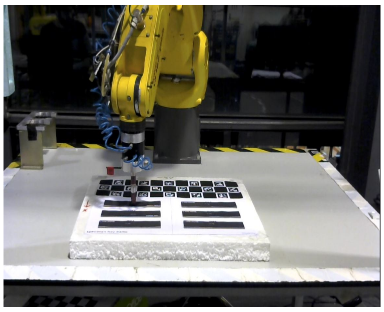
(b) Approach position 1



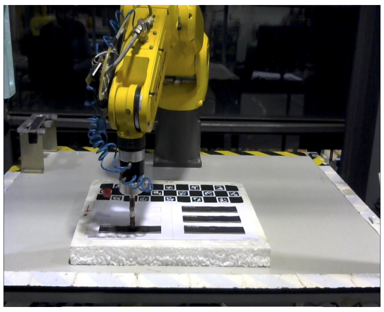
(c) Approach position 3



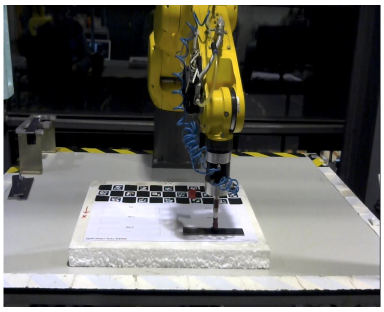
(d) Approach position 6



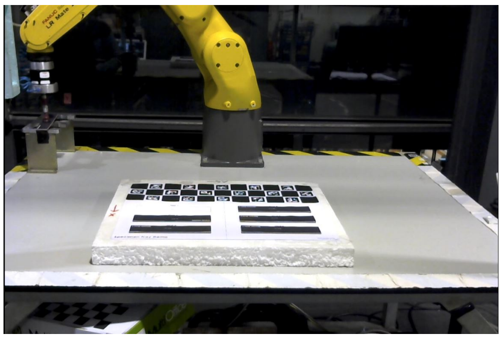
(e) Grasping point 1



(f) Grasping point 3



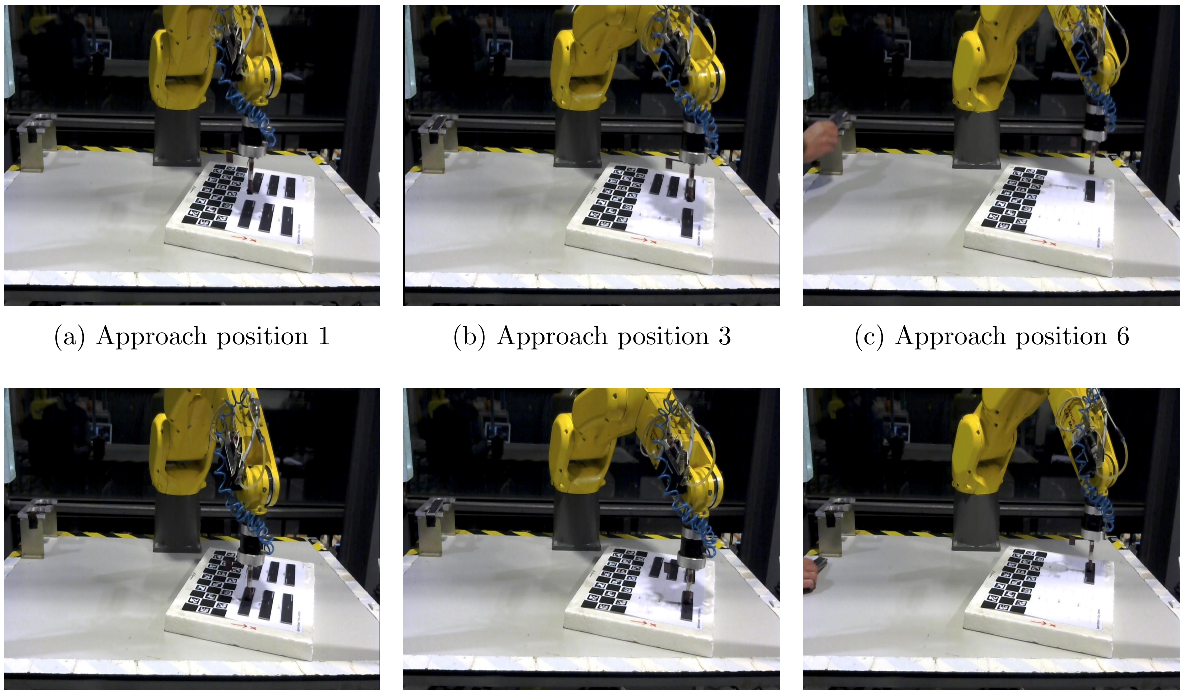
(g) Grasping point 6



(h) Destination point

Figure 10.4: Pick and Place results in Test 1.

Coordinate results for Test 2 tray's positions are presented on Table 10.2. The returned x and y coordinates, as well as the rotation components, show accurate results on the specimen picking spots (central points). The z coordinate, which is a fixed value of the tray for both tests, present small variations. These factors are analysed on Section 10.2. In the set of images from Figure 10.5, the representation of the Test 2 results are presented. Following the same concept from Test 1, Figures 10.5a 10.5b, 10.5c represent the approach points to specimens number 1, 3, and 6 of the tray, respectively. Figures 10.5d, 10.5e, and 10.5f show the grasping points to the same specimens.


(a) Approach position 1 (b) Approach position 3 (c) Approach position 6

(d) Grasping point 1 (e) Grasping point 3 (f) Grasping point 6

Figure 10.5: Specimen handling results in Test 2.

As an assistant option, this method/operation is available on video in `https://www.youtube.com/watch?v=qlHQGAHaHng&t=45s`, where the tray tracking process and specimen handling with the manipulator is shown.

Table 10.2: Specimen grasping point results for Test 2 tray's position.

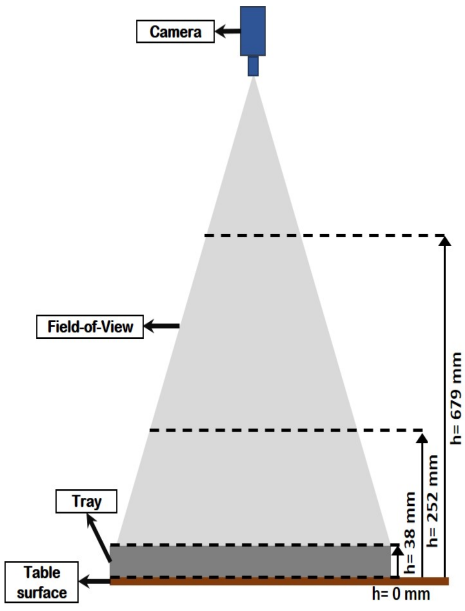
Coord		**x (mm)**	**y (mm)**	**z (mm)**	**Ψ (º)**	**θ (º)**	**φ(º)**
Specimen Index	**0**	550.241	190.59	31.3944	179.585	-0.383198	158.738
	1	570.188	241.844	31.7925	179.585	-0.383198	158.738
	2	590.135	293.098	32.1906	179.585	-0.383198	158.738
	3	345.221	270.368	32.8658	179.585	-0.383198	158.738
	4	365.168	321.622	33.2639	179.585	-0.383198	158.738
	5	385.115	372.876	33.662	179.585	-0.383198	158.738

10.2 Cases Discussion

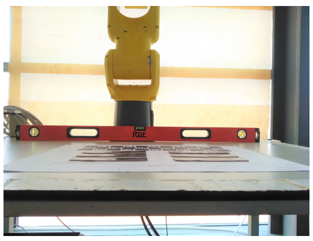
This section discusses the results of tests performed at different tray heights and tray occlusion conditions.

z coordinate precision

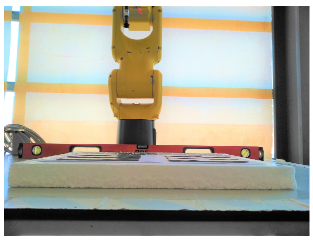
As represented on the layout of Figure 10.6, several tests were performed positioning the tray at different heights. The aim is to analyse the differences between the z coordinate values (returned by the detection code) and the real value (reference height) in each case. Table 10.3 contains the z coordinates of the different height returned by the tests.



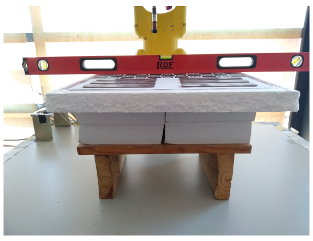
(a) Layout



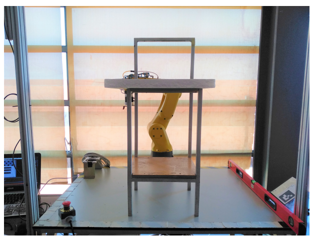
(b) h= 0 mm



(c) h= 38 mm



(d) h= 252 mm



(e) h= 679 mm

Figure 10.6: Different heights of the specimens tray tested.

Table 10.3: Z coordinate results for the different tray heights

		Z coordinate (mm)			
		h=0	**h=38**	**h=252**	**h=679**
Specimen Slot	**1**	-13.9434	26.0902	250.708	685.657
	2	-13.8682	25.4338	249.417	685.619
	3	-13.793	24.7774	248.126	685.581
	4	-11.6292	28.5945	254.171	686.848
	5	-11.554	27.9381	252.88	686.81
	6	-11.4788	27.2817	251.589	686.771

Figure 10.7 presents the absolute error of each specimen grasping points (between z

coordinates returned and z real values). This analysis aims to quantify the error at different heights. Positioning the tray at 252 mm (Figure 10.6d) is the case in which the error is the lowest. As represented by the results returned of 0 mm and 38 mm heights, the absolute error rate increases for lower heights (which corresponds to higher distances of the tray to the camera). Moreover, it was expected that the tray's proximity to the camera could be a factor of more accuracy in z coordinate results. That was not the case here, where the 680 mm tray's height return higher values of error than the 252 mm height case.

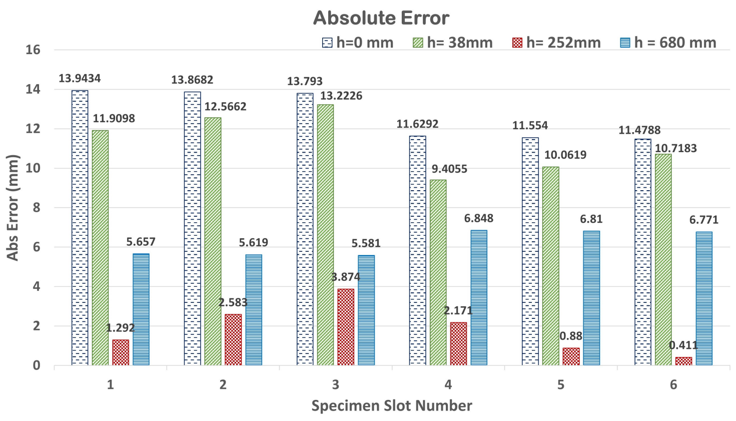


Figure 10.7: The bar chart analysis expresses the error values for different tray heights (Absolute Error).

Figure 10.8 expresses the normal difference between z coordinates returned by the code and the real heights. It aims to characterize the type of error (its trend). In this practical case, positive error values will place the suction grip (end-effector) to grasp specimens above the real picking region (before it reaches the tray). On the other hand, z coordinates presenting negative values of error can cause damages on the tray, because the grasping region is below the tray's surface. Note that closer distances of the tray to the camera produces z coordinates with positive margins of error (before reaching the whole tray/safe positions), and vice-versa for longer distances.

Several reasons can be the origin of these errors. One of them can be the rough surface caused by sticking the layout model sheet to the structure selected for the tray's prototype. This slight unevenness in the board's markers can cause the errors while the code is detecting/interpolating the references of the tray. Another source of error may appear in the process of matching perfectly (trying) both Aruco (from tool calibrator) and end-effector frames in Hand-Eye calibration. However, these error rates did not stop the overall suction grip operation. Performing this picking task using this type of end-effector does not require full contact with the object to grasp it, but can cause a small deviation in specimen central points grasped.

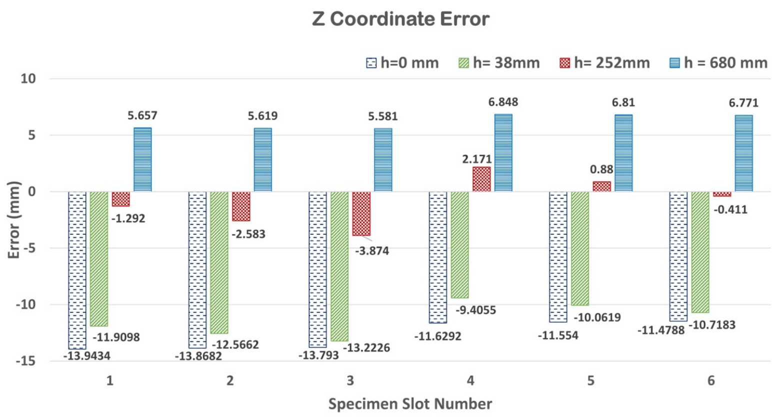
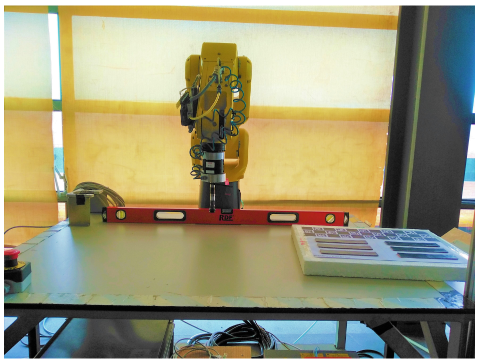


Figure 10.8: Normal difference between the reference heights and each z-coordinate returned.

Tray's occlusion

One of the points validated on the Charuco mark features is the detection of its board with occlusion. In this practical work, examples of this detection type (and the consequent return of the reference) are presented when the tray is (i) partially outside of the camera's FOV (Figure 10.9), (ii) has its centre occluded by the manipulator's arm (Figure 10.10a) or (iii) its reference (Figure 10.10b). However, if the occlusion rate is high on the tray (Figure


(a) Workspace view.

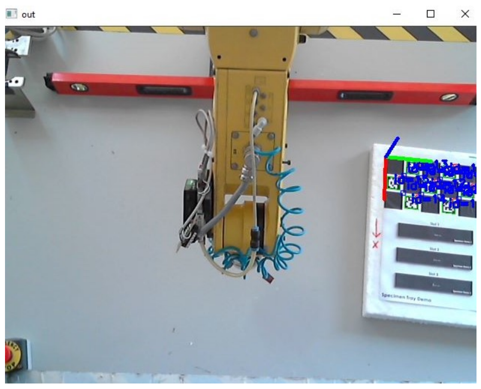

(b) Detection in VS.

Figure 10.9: Specimen tray outside of camera's FOV.

10.10c), the detection does not succeed. This feature is suitable to plan trajectories of the manipulator's arm with more versatility or fewer constraints on camera's occlusion.

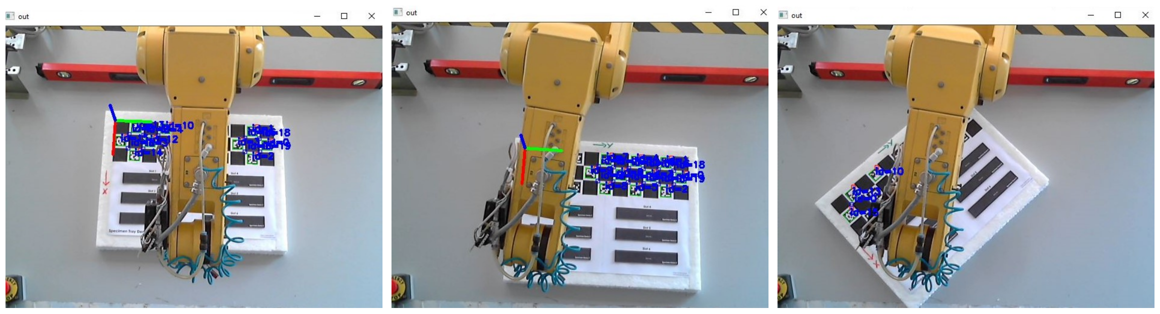

(a) (b) (c)

Figure 10.10: Board occlusion conditions: Arm (a) covering the center, (b) the reference , and (c) occluding almost entirely the board.

Part VII

Final Remarks

Chapter 11

Conclusions and Further Works

11.1 Conclusions

In this work, an experimental communication framework was projected, aiming to accomplish specimen handling in automated material testing. The contributions made were performed on the layout design apparatus created and within some specific modules, including (i) Testing Machine, (ii) Robotic, (iii) Visual Perception, and (iv) Control Modules.

In the Testing Machine Module, a biaxial testing machine (adjusted to perform tensile testing) was selected to retrofit some hardware/software features in it, which increases the degree of automation and versatility, in order to enable further integration on a automated mechanical testing system. Here, electrical switches were appended to control the grips courses, the homing procedure implemented, and the motor tracking positions as well.

On the Robotic Module, the TCP/IP Server was configured in order to establish a proper connection with LabVIEW. Here, a licensed library was provided by courtesy of Digimetrix, including coding blocks capable of interacting with Fanuc LR Mate 200 iD manipulator through this environment. Therefore, a Pick and Place routine is created, ready to receive the coordinates of the specimen positions in the tray from the Visual Perception Module.

Regarding this dissertation's objectives, the features integrated into the Visual Perception Module were the most satisfactorily and the achievements were able to be validated. The entire calibration procedure was performed via ROS Industrial due to its versatility and ease of integration. Furthermore, these results were imported to a Charuco detection algorithm, which tracks the specimen tray reference and returns the cartesian coordinates of each specimen within it to perform further grasping and handling tasks using the manipulator's gripper. This algorithm can be applied to future specimen trays/stations designed for these purposes. The requirements for potential components designed are (i) to have the Charuco marker integrated, and (ii) to know the fixed distances between each specimen centre points and this reference marker.

Robotic and Visual Perception modules are integrated into the LabVIEW environment for testing purposes. The gripper used to handle the specimens is the suction grip initially coupled to the manipulator's end-effector, which is suitable to handle the specimens in these tests. Here again, the gripper format depends on future specimen trays designed and, if required, can be modified/re-designed according to its demands.

On the other hand, the Testing Machine Module has not achieved the integration stage with these two modules yet, not being able to test the new automated features added to the machine in the integrated environment.

Given the unavailability to have the Fanuc robot and testing machine within the same laboratory, a specimen tray prototype was designed in the practical case. After that, the two modules (Robotic and Visual Perception) integrated are tested within the workspace. The specimen handling operation aids to validate the robotic pick and place routine created, and the coordinates returned by the Visual Perception algorithm.

The ROS packages used for calibrations and the working system developed for this dissertation are publicly available on:

- Msc_TesteAutomation
  - `https://github.com/diegosaraivanunes/Msc_TesteAutomation`;
- Camera Calibration (Intrinsics)
  - `https://www.youtube.com/watch?v=QEqIVUfse30`;
- Hand-Eye Calibration (Extrinsic)
  - `https://www.youtube.com/watch?v=0MHRzPrFhzg&t=198s`;
- Tray Tracking and Specimen handling
  - `https://www.youtube.com/watch?v=qlHQGAHaHng&t=45s`.

11.2 Future Works

This dissertation is integrated on the main research group project named as "Integration of Robotic Systems in Material Testing". This work was the first approach to this thematics (kick-off relatively to practical cases) and contributed to what is aiming to accomplish as the final goal: fully automatic testing. Several modules must be integrated and features added in the future. In particular, by priority order, they are listed above:

- Add pneumatic/hydraulic grips to the testing machine module to allow the remote control;
- Enable the data acquisition of the load cells to collect strain/stress information during testing;
- Integrate the testing machine features retrofitted in this work to the automatic testing;
- Append the data recording module (Video-Extensometer or Aramis) to register the testing results;
- Design and produce a real specimen tray/station to verify detection algorithm effectiveness;
- Transport manipulator and testing machine to the same lab in order to proceed to the final system integration, and consequently, fully automatic testing.

Bibliography

[1] Meredith Platt and Jim Ritchey. Sources of error in mechanical testing results. https://bit.ly/2PydKTa, 2007. [Online; accessed 31-January-2020].

[2] Norman E. Dowling. *Mechanical Behavior of Materials.* Pearson Education Limited, 2012.

[3] Sundar Dannana. What are the different material testing methods? (testing of materials). https://bit.ly/306YxfU, 2018. [Online; accessed 10-January-2020].

[4] victoradmin. Mechanical testing of composites. https://www.victortestingmachine.com/why-use-tensile-test/, 2019. [Online; accessed 25-January-2020].

[5] J. R. Davis. *Tensile Testing.* ASM International, 2nd edition, 2004.

[6] Deniz Yalcin. digitalimagecorrelation.org. https://www.admet.com/measurement-errors-in-material-testing/, 2019. [Online; accessed 31-January-2020].

[7] YH Wang, JH Jiang, C Wanintrudal, C Du, D Zhou, LM Smith, and LX Yang. Whole field sheet-metal tensile test using digital image correlation. *Experimental Techniques*, 34(2):54–59, 2010.

[8] IF of Robotics. Executive summary world robotics 2020 industrial robots. *World Robotics 2020 edition*, pages p.13–16, 2020.

[9] Alessandro Selvaggio, Abdelhakim Sadiki, Tobias R Ortelt, Rickmer Meya, Christoph Becker, Sami Chatti, and A Erman Tekkaya. Development of a cupping test in remote laboratories for engineering education. In *Engineering Education 4.0*, pages 465–476. Springer, 2016.

[10] Abdelhakim Sadiki, Tobias R Ortelt, Christian Pleul, Christoph Becker, Sami Chatti, and A Erman Tekkaya. The challenge of specimen handling in remote laboratories for engineering education. In *Proceedings of 2015 12th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, pages 180–185. IEEE, 2015.

[11] Tobias R Ortelt, Abdelhakim Sadiki, Christian Pleul, Christoph Becker, Sami Chatti, and A Erman Tekkaya. Development of a tele-operative testing cell as a remote lab for material characterization. In *Engineering Education 4.0*, pages 265–277. Springer, 2016.

[12] ZwickRoell. Automated Testing Systems . https://www.zwickroell.com/en/automated-testing-systems. [Online; accessed 28-January-2020].

[13] Instron. Automated Testing Systems. `https://www.instron.co.hu/hu-hu/products/testing-systems/automated-testing-systems`. [Online; accessed 28-January-2020].

[14] Hu Guolin. Full-automatic tension testing machine. [Online]. Available from: `https://patents.google.com/patent/CN102778393A/en?oq=CN102778393A`, November 14 2012. CN Patent CN102778393A.

[15] Wang Yiyun. Tension tester and sample placing device thereof. [Online]. Available from: `https://patents.google.com/patent/CN202649029U/en?oq=CN102778393A`, January 02 2013. CN Patent CN202649029U.

[16] Dan Lu. Mechanical hand device for automatically clamping and discharging sample up and down for testing machine. [Online]. Available from: `https://patents.google.com/patent/CN101598736A/en?oq=CN102778393A`, December 09 2009. CN Patent CN101598736A.

[17] Xu Weicheng. Automatic sample unloading mechanical hand device for tensile test machine. [Online]. Available from: `https://patents.google.com/patent/CN105571937A/en?oq=CN105571937a`, March 05 2016. CN Patent CN105571937A.

[18] Han Bin. Automatic test specimen sorting device of full-automatic tensile testing machine. [Online]. Available from: `https://patents.google.com/patent/CN203561545U/en?oq=CN105571937a`, March 04 2014. CN Patent CN203561545U.

[19] Chen Guangfeng. Automatic feed manipulator device with fiber tension tester. [Online]. Available from: `https://patents.google.com/patent/CN203726494U/en?oq=cn203726494u`, March 07 2014. CN Patent CN203726494U.

[20] João PM Maio. *Projeto e Construção de Máquina de Ensaios Biaxiais*. PhD thesis, Dissertação de Mestrado Integrado em Engenharia Mecânica, Universidade de Aveiro, 2014.

[21] Fanuc. Datasheet LR Mate 200iD. `https://bit.ly/2Iv9WBw`. [Online; accessed 08-April-2020].

[22] Fanuc. FANUC R-30iB Open Air Cabinet Controller. `https://bit.ly/2IxSBID`. [Online; accessed 26-October-2020].

[23] AaronMR. ROS Concepts. `http://wiki.ros.org`. [Online; accessed 15-November-2020].

[24] Robert John. Hands-On Introduction to Robot Operating System(ROS). `https://bit.ly/38N5J7g`. [Online; accessed 15-November-2020].

[25] DigiMetrix. Fanuc robotics library. `http://www.digimetrix.com/products/fanuc/`. [Online; accessed 30-October-2020].

[26] EA Ruth. Elements of automated mechanical testing. In *Automation of Mechanical Testing*. ASTM International, 1993.

[27] Shimadzu. Fully Automatic Material Testing Systems. `https://www.shimadzu.com`. [Online; accessed 27-January-2020].

[28] Malcolm S Loveday, Tom Gray, and Johannes Aegerter. Tensile testing of metallic materials: A review. *Final report of the TENSTAND project of work package*, 1, 2004.

[29] José Aquino, A Gil Andrade-Campos, João MP Martins, and Sandrine Thuillier. Design of heterogeneous mechanical tests: Numerical methodology and experimental validation. *Strain*, 55(4):e12313, 2019.

[30] N Souto, A Andrade-Campos, and S Thuillier. Mechanical design of a heterogeneous test for material parameters identification. *International Journal of Material Forming*, 10(3):353–367, 2017.

[31] António Andrade-Campos, José Aquino, João MP Martins, and Bernardete Coelho. On the design of innovative heterogeneous sheet metal tests using a shape optimization approach. *Metals*, 9(3):371, 2019.

[32] Juan Zhang, Ahmed Sweedy, François Gitzhofer, and Gamal Baroud. A novel method for repeatedly generating speckle patterns used in digital image correlation. *Optics and Lasers in Engineering*, 100:259–266, 2018.

[33] Arduino.cc. Arduino Uno Rev3. `https://store.arduino.cc/arduino-uno-rev3`. [Online; accessed 16-October-2020].

[34] Logitech. C270 HD WEBCAM. `https://www.logitech.com/pt-br/product/hd-webcam-c270`. [Online; accessed 16-October-2020].

[35] Robot Operating System (ROS). ROS-Industrial. `https://rosindustrial.org`. [Online; accessed 16-October-2020].

[36] National Instruments. labVIEW software. `https://www.ni.com/pt-pt/shop/labview.html`. [Online; accessed 15-October-2020].

[37] N Saba, M Jawaid, and MTH Sultan. An overview of mechanical and physical testing of composite materials. In *Mechanical and Physical Testing of Biocomposites, Fibre-Reinforced Composites and Hybrid Composites*, pages 1–12. Elsevier, 2019.

[38] Ian McEnteggart. Mechanical testing of composites. `https://bit.ly/2Gmuxnm`, 2014. [Online; accessed 24-January-2020].

[39] Sundar Dannana. How compression test is conducted? `https://bit.ly/2uxID2v`, 2018. [Online; accessed 26-January-2020].

[40] Dong-Joong Kang, Jun-Hyub Park, Myung-Soo Shin, Jong-Eun Ha, and Hak-Joo Lee. Specimen alignment in an axial tensile test of thin films using direct imaging and its influence on the mechanical properties of becu. *Journal of Micromechanics and Microengineering*, 20(8):085001, 2010.

[41] Leonardo Martinez. Why measure strain optically? `https://bit.ly/316uXYA`, 2015. [Online; accessed 31-January-2020].

[42] Will LePage. digitalimagecorrelation.org. `https://digitalimagecorrelation.org/`, 2017. [Online; accessed 31-January-2020].

[43] P Gebhardt. Experiences in the automation of mechanical testing. In *Automation of Mechanical Testing*. ASTM International, 1993.

[44] Earl Ruth. Robotic materials testing systems come of age. *Quality*, 31(11):31, 1992.

[45] AZOMaterials. Automatic Materials Testing Machines. `https://www.azom.com/materials-equipment.aspx?cat=45`. [Online; accessed 28-January-2020].

[46] Shimadzu. Automatic Systems. `https://www.shimadzu.com/an/test/universal/index.html#auto`. [Online; accessed 09-April-2020].

[47] Steppermotoronline. Stepper Motor Driver MA860H 2.4-7.2A Max 80VAC or 110VDC. `https://bit.ly/37B2DTs3`. [Online; accessed 25-October-2020].

[48] Shimadzu. Specifications of AG-Xplus floor type. `https://www.shimadzu.eu/specifications-ag-xplus-floor-type`. [Online; accessed 08-April-2020].

[49] Shimadzu. Hydraulic Non-Shift Wedge Grips. `https://bit.ly/331n9tb`. [Online; accessed 08-April-2020].

[50] Shimadzu. Pneumatic Non-Shift Wedge Grips. `https://bit.ly/35K74K5`. [Online; accessed 08-April-2020].

[51] Rui Cancela. Extensão e flexibilização da interface de controlo de um manipulador robótico fanuc, 2007.

[52] Miguel Riem Oliveira. Ros workshop. `https://github.com/miguelriemoliveira/rws2020_moliveira/wiki`. [Online; accessed 28-February-2020].

[53] Open Source Robotics Foundation. Ros tutorials. `http://wiki.ros.org/ROS/Tutorials`. [Online; accessed 28-February-2020].

[54] Open Robotics Foundation. Open Robotics. `https://www.openrobotics.org`. [Online; accessed 16-November-2020].

[55] Open Robotics. ROS. `https://www.ros.org`. [Online; accessed 12-February-2020].

[56] G. vd. Hoorn. ROS Fanuc Tutorials. `http://wiki.ros.org/fanuc/Tutorials`. [Online; accessed 15-November-2020].

[57] G. vd. Hoorn. fanuc drivers. `https://github.com/ros-industrial/fanuc`. [Online; accessed 17-November-2020].

[58] ROS-Industrial. Fanuc experimental. `https://github.com/ros-industrial/fanuc_experimental`. [Online; accessed 23-November-2020].

[59] PickNik Robotics. Moveit. `https://moveit.ros.org`. [Online; accessed 23-November-2020].

[60] Dassault Systemes. Solidworks solutions. `https://www.solidworks.com/solutions`. [Online; accessed 28-February-2020].

[61] Franka Emika. Moveit setup assistant. `https://bit.ly/375hF1N`. [Online; accessed 23-November-2020].

[62] Tiago F Simões. *Integração de ROS Industrial num robo FANUC para flexibilizar atividades de cooperação*. PhD thesis, Dissertação de Mestrado Integrado em Engenharia Mecânica, Universidade de Aveiro, 2016.

[63] Microsoft. Visual Studio Community 2019. `https://visualstudio.microsoft.com/vs/community/`. [Online; accessed 19-November-2020].

[64] OpenCV. opencv-master. `https://github.com/opencv/opencv`. [Online; accessed 19-November-2020].

[65] OpenCV. opencv-contrib. `https://github.com/opencv/opencv_contrib`. [Online; accessed 19-November-2020].

[66] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.

[67] OpenCV. Detection of charuco corners. `https://docs.opencv.org/3.4/df/d4a/tutorial_charuco_detection.html`. [Online; accessed 23-November-2020].

[68] Inc. The MathWorks. What is camera calibration? https://www.mathworks.com/help/vision/ug/camera-calibration.html. [Online; accessed 23-November-2020].

[69] ROS wiki. How to calibrate a monocular camera. http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration. [Online; accessed 23-November-2020].

[70] Vitor M. F. Santos. *Robotica Industrial - Apontamentos Teoricos*. Departamento de Engenharia Mecanica, 2004.

[71] ROS Planning. Moveit calibration. https://github.com/ros-planning/moveit_calibration. [Online; accessed 01-October-2020].

[72] Wiki ROS. Hand-eye calibration. http://wiki.ros.org/rc_visard/Tutorials/HandEyeCalibration. [Online; accessed 01-October-2020].

[73] Franka Emika. Hand-eye calibration tutorial. https://bit.ly/3mc7vTg. [Online; accessed 01-October-2020].

[74] Konstantinos Daniilidis. Hand-eye calibration using dual quaternions. *The International Journal of Robotics Research*, 18(3):286–298, 1999.

[75] OpenCV. Detection of charuco corners. https://docs.opencv.org/3.4/df/d4a/tutorial_charuco_detection.html. [Online; accessed 30-October-2020].

[76] OpenCV. Camera calibration and 3d reconstruction. https://bit.ly/2Jph0Ak. [Online; accessed 27-November-2020].

[77] Satya Mallick. Rotation matrix to euler angles function examples. https://www.learnopencv.com/rotation-matrix-to-euler-angles/. [Online; accessed 28-October-2020].

Part VIII

Appendices

Appendix A

Requested Quotes

Certificate Registration No. 510472

Izasa Scientific Lda
SEDE: *Edifício Ramazzotti, Av. do Forte, 6, Piso 3,*
Porta 2.24 2790-072 CARNAXIDE
Tel.: 214 247 300 - Fax: 214 255 283
NORTE:*Rua Manuel Salgueiral, n.º 424*
4400-213 Vila Nova de Gaia
Tel.: 223 778 400 - Fax 223 778 490

N/Referência	Proposta	Data	V/Referência
	31959102	**27.03.2020**	

HYDRAULIC NON-SHIFT WEDGE T. GRIPS 100KN
Referência: 536343-04971-11

Preço Total: EUR 10.970,00
dez mil novecentos e setenta euros

BOMBA HIDRÁULICA DE 4 VÁLVULAS
Apresentação: UNID
Referência: 536TH135-P2-2-G-A

Preço Total: EUR 5.015,00
cinco mil e quinze euros

COMANDO REMOTO
Apresentação: UNID
Referência: 536TH188-HSW

Preço Total: EUR 1.254,00
mil duzentos e cinquenta e quatro euros

INSTALAÇÃO
Referência:

Preço Total: EUR 500,00
quinhentos euros

PRAZO DE ENTREGA:	Dependente do stock existente
GARANTIA:	Um ano contra defeitos de fabrico ou de funcionamento
PORTES E EMBALAGEM:	Incluidos
IMPOSTO (IVA):	Não incluido, à taxa de 23%
FORMA DE PAGAMENTO:	A 30 dias Os pagamentos devem ser efetuados exclusivamente por transferência bancária, para a nossa conta no BBVA, Agência da Sede - Lisboa, com o IBAN PT50 0019 0866 0020 0004 3413 7
VALIDADE DA PROPOSTA:	90 dias

Izasa Scientific
A Werfen Company

Certificate Registration No. 510472

Izasa Scientific Lda
SEDE: *Edifício Ramazzotti, Av. do Forte, 6, Piso 3,*
Porta 2.24 2790-072 CARNAXIDE
Tel.: 214 247 300 - Fax: 214 255 283
NORTE: *Rua Manuel Salgueiral, n.º 424*
4400-213 Vila Nova de Gaia
Tel.: 223 778 400 - Fax 223 778 490

N/Referência	Proposta	Data	V/Referência
	31986273	**11.09.2020**	

LabView driver for AGS-X or AG-X

Apresentação:	UNID
Referência:	536SSOI3XB710132
Preço Total:	EUR 3.847,50
	tres mil oitocentos e quarenta e sete euros e cinquenta cent.

PRAZO DE ENTREGA:	60 dias
GARANTIA:	Um ano contra defeitos de fabrico
PORTES E EMBALAGEM:	Incluidos
IMPOSTO (IVA):	Não incluido, à taxa de 23%
FORMA DE PAGAMENTO:	A 30 dias Os pagamentos devem ser efetuados exclusivamente por transferência bancária, para a nossa conta no BBVA, Agência da Sede - Lisboa, com o IBAN PT50 0019 0866 0020 0004 3413 7

VALIDADE DA PROPOSTA:	90 dias

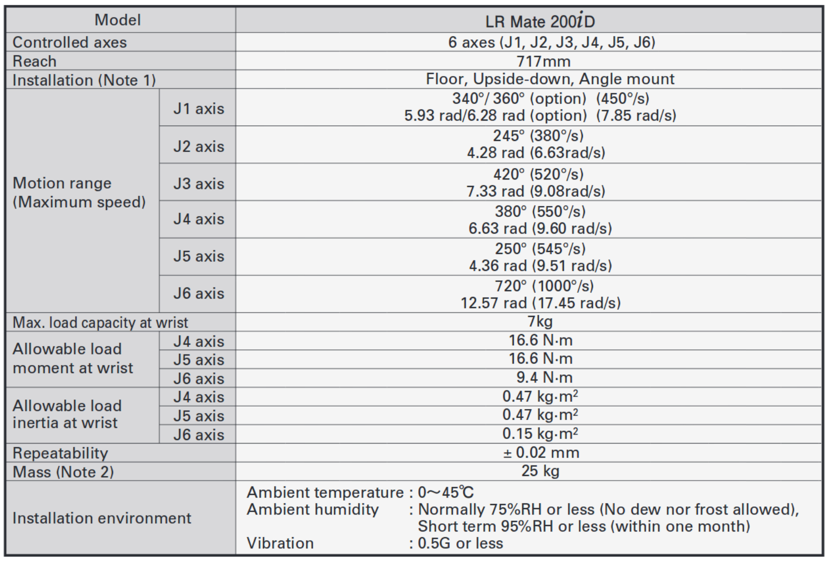
Lisboa, 11 de Setembro de 2020

Izasa Scientific
A Werfen Company

[illegible]

Appendix B

FANUC LR Mate 200iD specifications



Model		LR Mate 200*i*D
Controlled axes		6 axes (J1, J2, J3, J4, J5, J6)
Reach		717mm
Installation (Note 1)		Floor, Upside-down, Angle mount
Motion range (Maximum speed)	J1 axis	340°/ 360° (option) (450°/s) 5.93 rad/6.28 rad (option) (7.85 rad/s)
	J2 axis	245° (380°/s) 4.28 rad (6.63rad/s)
	J3 axis	420° (520°/s) 7.33 rad (9.08rad/s)
	J4 axis	380° (550°/s) 6.63 rad (9.60 rad/s)
	J5 axis	250° (545°/s) 4.36 rad (9.51 rad/s)
	J6 axis	720° (1000°/s) 12.57 rad (17.45 rad/s)
Max. load capacity at wrist		7kg
Allowable load moment at wrist	J4 axis	16.6 N·m
	J5 axis	16.6 N·m
	J6 axis	9.4 N·m
Allowable load inertia at wrist	J4 axis	0.47 kg·m^2
	J5 axis	0.47 kg·m^2
	J6 axis	0.15 kg·m^2
Repeatability		± 0.02 mm
Mass (Note 2)		25 kg
Installation environment		Ambient temperature : 0～45℃ Ambient humidity : Normally 75%RH or less (No dew nor frost allowed), Short term 95%RH or less (within one month) Vibration : 0.5G or less

Figure B.1: Technical specifications of the Fanuc LR Mate 200iD [21]

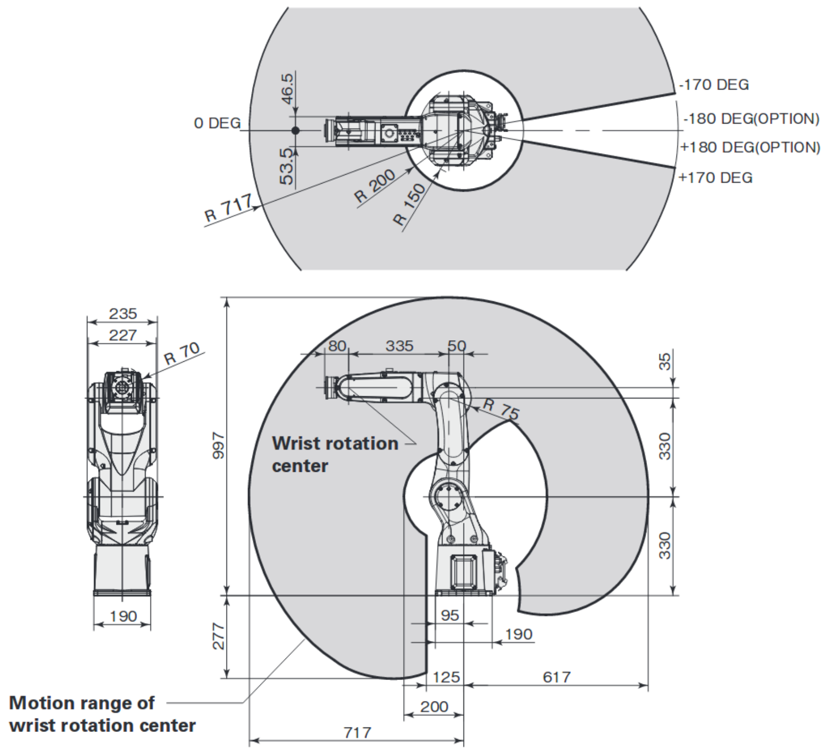
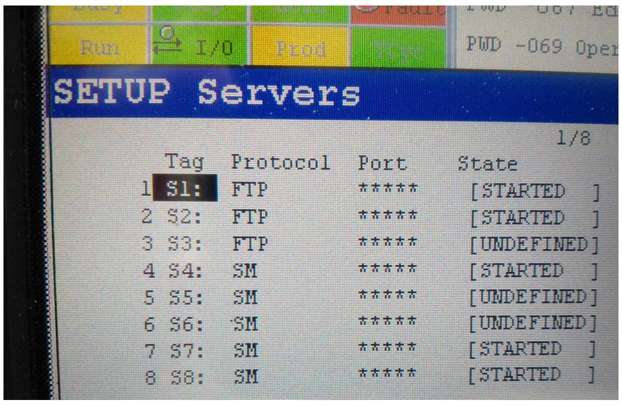


Figure B.2: Dimensions and working range of the manipulator [21].

Appendix C

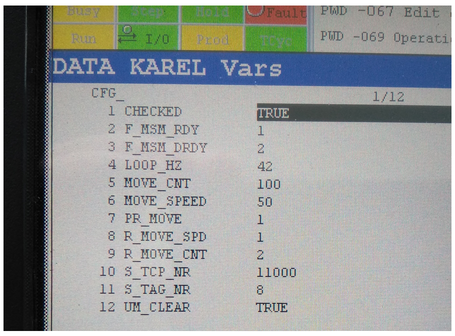
ROS Server Configurations

The configuration of the communication protocol for the ROS Server requires the execution of two FTP files coming from Cancela's [51] work and two new USM files. In Figure C.1a, the teach pendant view shows the required tag files with the protocol FTP started by default (S1 and S2), and the USM files (tags S7 and S8).



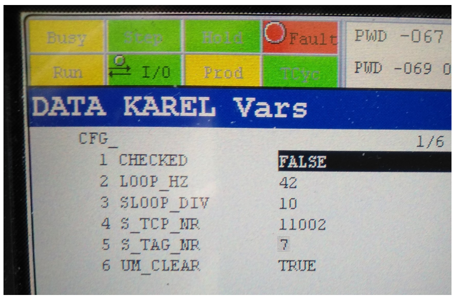
```
SETUP Servers
                                        1/8
    Tag  Protocol  Port    State
  1 S1:  FTP       *****   [STARTED   ]
  2 S2:  FTP       *****   [STARTED   ]
  3 S3:  FTP       *****   [UNDEFINED]
  4 S4:  SM        *****   [STARTED   ]
  5 S5:  SM        *****   [UNDEFINED]
  6 S6:  SM        *****   [UNDEFINED]
  7 S7:  SM        *****   [STARTED   ]
  8 S8:  SM        *****   [STARTED   ]
```

(a)



```
DATA KAREL Vars
  CFG_                            1/12
   1 CHECKED        TRUE
   2 F_MSM_RDY      1
   3 F_MSM_DRDY     2
   4 LOOP_HZ        42
   5 MOVE_CNT       100
   6 MOVE_SPEED     50
   7 PR_MOVE        1
   8 R_MOVE_SPD     1
   9 R_MOVE_CNT     2
  10 S_TCP_NR       11000
  11 S_TAG_NR       8
  12 UM_CLEAR       TRUE
```

(b)



```
DATA KAREL Vars
  CFG_                            1/6
   1 CHECKED        FALSE
   2 LOOP_HZ        42
   3 SLOOP_DIV      10
   4 S_TCP_NR       11002
   5 S_TAG_NR       7
   6 UM_CLEAR       TRUE
```

(c)

Figure C.1: (a) Configurations to execute setup Servers , (b) *ros_relay*, and (c) *ros_state* variables configurations presented in the teach pendant display.

The four main files generated in Roboguide compilation, and consequently transferred to the controller are (i) *ros_state.sm*[1], (ii) *ros_relay.sm*[2], (iii) *ros_movesm.tp*[3] and (iv) *ros.tp*[4].

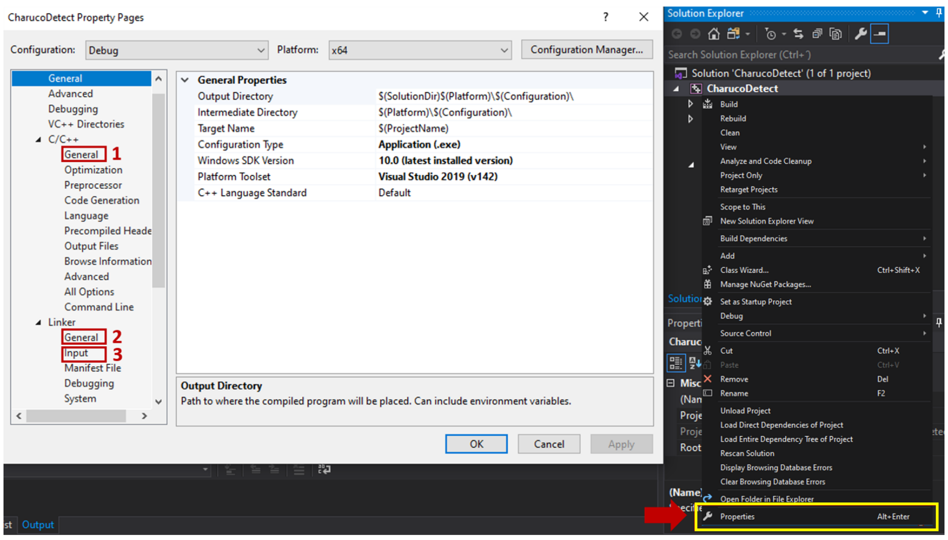
In addition, the pattern values for the files transferred to the controller (available on [56] tutorials) are defined in the variables of the *ros_relay* (Figure C.1b) and *ros_state* (Figure C.1c) programs.

[1]ROS server which is waiting in a pre-defined port to the client connection.
[2]ROS server responsible to receive the joint states of the manipulator.
[3]TP file program responsible to perform the manipulator's movements.
[4]Unique TP file executed by the user. It runs the three programs described above simultaneously.

Appendix D

OpenCV configurations in VS

The Visual Studio Community 2019 (VS) is executed on Microsoft Windows 10 operating system. Four main steps are required to configure the OpenCV libraries: (1) Define the path in system variables, (2) add additional include directory, (3) additional library directories, and (4) additional dependencies. Only the first step is configured in the system environment of the operating system (Environment Variables). The others are defined inside the properties of the Visual Studio project created. The configuration of the VS project is always set as Debug mode and in an x64 platform. The content from *opencv_contrib* [65] repository is extracted to the main directory C, within a folder named as *opencv_lib*. Next, the steps referred above are described.


CharucoDetect Property Pages

Configuration: Debug Platform: x64

General Properties

Output Directory	$(SolutionDir)$(Platform)\$(Configuration)\
Intermediate Directory	$(Platform)\$(Configuration)\
Target Name	$(ProjectName)
Configuration Type	**Application (.exe)**
Windows SDK Version	**10.0 (latest installed version)**
Platform Toolset	**Visual Studio 2019 (v142)**
C++ Language Standard	Default

Output Directory
Path to where the compiled program will be placed. Can include environment variables.

Figure D.1: Properties of the project.

Define path in system variables

In Windows *System properties* → *Environment Variables* environment, the following path of the system variables is created `C:\opencv_lib\install\x64\vc16\bin`

Add additional include directory

Inside the VS project containing *Charuco detection* algorithm, in its properties (yellow rectangle pointed out in Figure D.1), the include path in C++ (red rectangle 1 on Figure D.1) is set with the path `C:\opencv_lib\install\include`.

Add additional library directories

In project properties (yellow rectangle on Figure D.1), inside *Linker* section (red rectangle 2), the environmental library path is defined on *Additional Library Directories* with `C:\opencv_lib\install\x64\vc16\lib` directory.

Add additional dependencies

The additional library dependencies required for this project are set on *Linker*→*Input* (red rectangle 3 on Figure D.1). The list of library files (Debug-version) added as dependencies to the project is presented below:

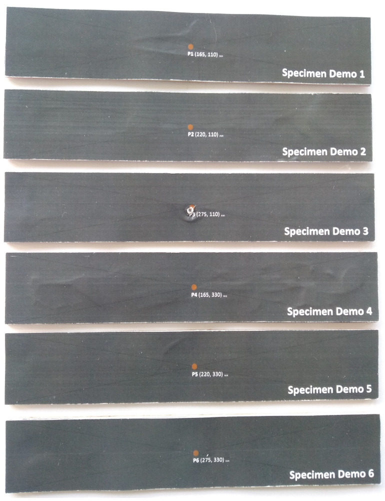
- *opencv_aruco*450*d.lib*
- *opencv_calib*3*d*450*d.lib*
- *opencv_ccalib*450*d.lib*
- *opencv_core*450*d.lib*
- *opencv_highgui*450*d.lib*
- *opencv_imgcodec*450*d.lib*
- *opencv_imgproc*450*d.lib*
- *opencv_videoio*450*d.lib*
- *opencv_video*450*d.lib*

These libraries allow the use of functions and object classes from OpenCV required to the project. For example, the *opencv_aruco*450*d.lib* allows the use of Aruco class objects in programming in order to create the algorithm to detect the Charuco markers.

Appendix E

Specimens Tray Demo

The output of *Charuco detection* algorithm and consequent conversion to world coordinates (at robot base) is validated by the design of a specimen's tray prototype (including the specimens in Figure E.1a). The displacement between each grasping point/region (in each slot) and the tray origin defined by the Charuco reference must be known (available in a text file format) and ready to be loaded when the detection algorithm requires the import (Table E.1).



(a)

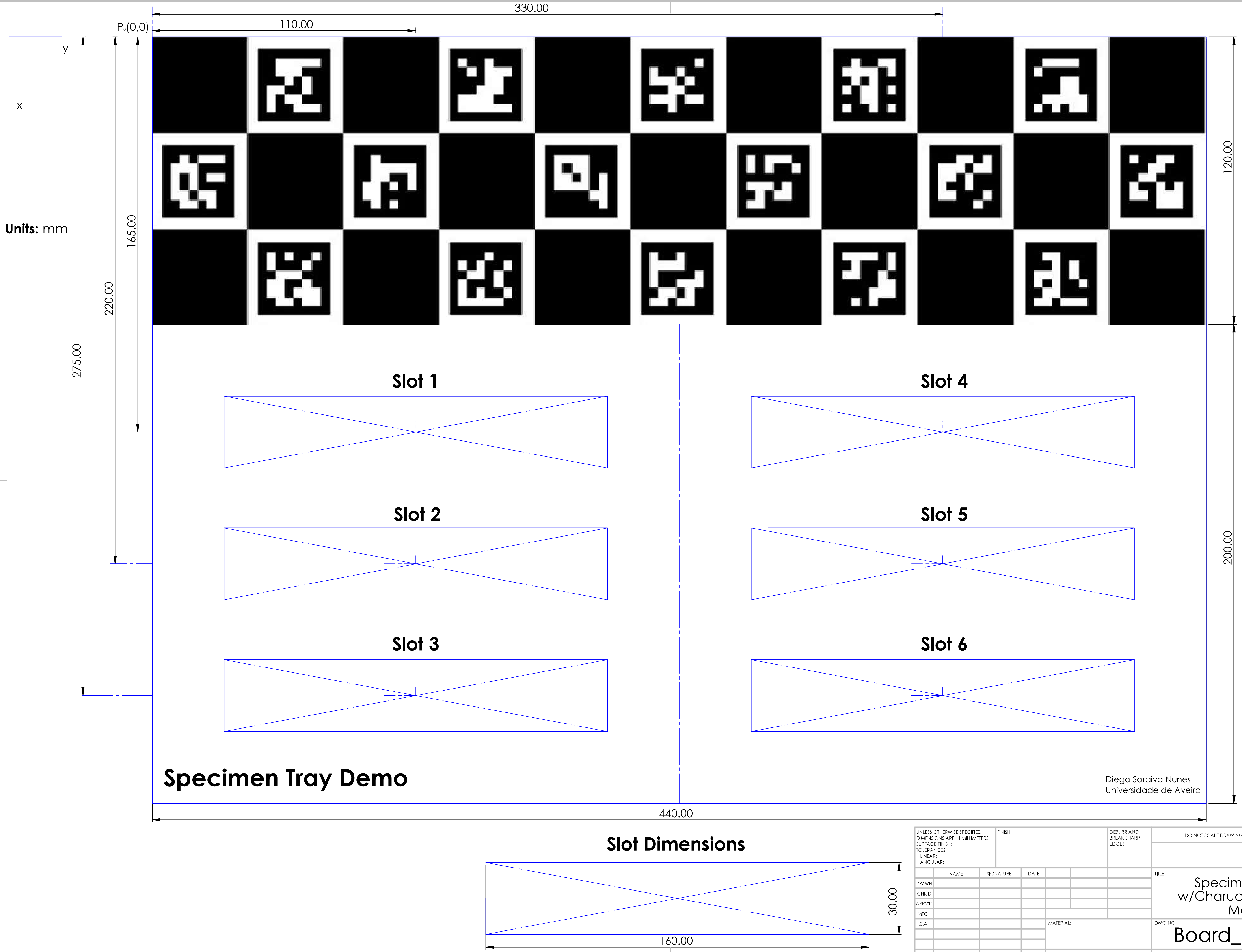
(b)

Figure E.1: (a) Prototypes of the specimens, and (b) their thickness.

The template of the tray's surface used (with all the dimensions included) is presented on the technical draw in the next page, including the dimensions of the specimen unit.

Table E.1: Grasping coordinates of each point to the origin of the tray/Charuco board reference (in *mm*).

Tag	**x**	**y**	**z**
P1	165	110	0
P2	220	110	0
P3	275	110	0
P4	165	330	0
P5	220	330	0
P6	275	330	0



Units: mm

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:	DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE	TITLE: Specimens Board w/Charuco Reference Marker
DRAWN				
CHK'D				
APPV'D				
MFG				
Q.A			MATERIAL:	DWG NO. Board_layout — A2
			WEIGHT:	SCALE:1:1 — SHEET 1 OF 1

Appendix F

Appends to Arduino Code

```
#include "Timer.h"

// 1) Declare variables and Arduino pins
  // Pulse
const int mv1 = 3; //OUTPUT pin - PUL+ (motor1)
const int mv3 = 6; //OUTPUT pin - PUL+ (motor3)
  // Direction
const int md1 = 2; // DIR+ (motor1)
const int md3 = 7; // DIR+ (motor3)
// 2) SWITCH pins
const int switchPin_lower = 10; // Limit Lower Switch pin
const int switchPin_upper = 11; // Limit Upper Switch pin
// 3) Positioning variables
float  pos_inf; // Positioning of inf motor (motor1)
float  pos_sup; // Positioning of sup motor (motor3)

// put your setup code here, to run once:
void setup() {
  // 4) Pin modes
  pinMode(mv1,OUTPUT);
  pinMode(mv3,OUTPUT);
  pinMode(md1,OUTPUT);
  pinMode(md3,OUTPUT);

  // 5) Define Starting status
  digitalWrite(mv1,LOW);
  digitalWrite(mv3,LOW);
  digitalWrite(md1,LOW);
  digitalWrite(md3,LOW);

  // 6) Limit switches Pin modes input pull_up  resitor
  pinMode(switchPin_lower,INPUT_PULLUP);
  pinMode(switchPin_upper,INPUT_PULLUP);

  Serial.begin(1000000);

  // 7) Start Homing procedure of Vertical Stepper Motors at startup:
  // 7.1) For lower switch
  while (digitalRead(switchPin_lower)) {  // Do this until the switch is not activated
    digitalWrite(md1, HIGH);      // (HIGH = anti-clockwise / LOW = clockwise) -
    dirPin_lower
    digitalWrite(mv1, HIGH);
    delayMicroseconds(500);                        // Delay to slow down speed of Stepper
    digitalWrite(mv1, LOW);
    delayMicroseconds(500);
}
  while (!digitalRead(switchPin_lower)) { // Do this until the switch is not activated
    digitalWrite(md1, LOW);
    digitalWrite(mv1, HIGH);
    delayMicroseconds(500);                      // More delay to slow even more while
    moving away from switch
    digitalWrite(mv1, LOW);
    delayMicroseconds(500);
}

  // 7.2) For upper switch
  while (digitalRead(switchPin_upper)) {  // Do this until the switch is activated
    digitalWrite(md3, HIGH);      // (HIGH = anti-clockwise / LOW = clockwise)
    digitalWrite(mv3, HIGH);
    delayMicroseconds(500);                       // Delay to slow down speed of Stepper
    digitalWrite(mv3, LOW);
    delayMicroseconds(500);
}
  while (!digitalRead(switchPin_upper)) { // Do this until the switch is not activated
    digitalWrite(md3, LOW);
    digitalWrite(mv3, HIGH);
    delayMicroseconds(500);                      // More delay to slow even more while
    moving away from switch
    digitalWrite(mv3, LOW);
    delayMicroseconds(500);
  }

// 8) Reset Positions variable to zero:
 pos_inf=0;
 pos_sup=0;
}
/// MAIN CODE
void loop() {
  // put your main code here, to run repeatedly:
  if (pos_inf>=0 && pos_inf<=100 && pos_sup>=0 && pos_sup<=85){ // Condition to provent
    courses off limits
             // ... Original Code ...
    }
}
```

Appendix G

LabVIEW modules specifications

This appendix specifies a modular part of the LabVIEW code built for the system.

G.1 Array-to-Cartesian SubVI

This modular sub-routine merges (in every iteration) the array of the coordinates imported to the remaining (and constant) fields, resulting in the *typedef* structure with the data format required for input data in manipulator's *Move Synchro* blocks. Table G.1 presents an example of the coordinates format file returned by a random position captured of the specimen tray, and Figure G.1 the block diagram of this conversion.

Table G.1: Example of an array with specimen grasping points returned.

		X (mm)	**Y (mm)**	**Z (mm)**	**W (°)**	**P (°)**	**R (°)**
Specimen Index	**0**	611.825	-257.991	28.9311	-179.946	-0.562395	91.5384
	1	666.805	-256.514	28.8789	-179.946	-0.562395	91.5384
	2	721.786	-255.037	28.8268	-179.946	-0.562395	91.5384
	3	605.919	-38.0812	31.0905	-179.946	-0.562395	91.5384
	4	660.899	-36.6041	31.0384	-179.946	-0.562395	91.5384
	5	715.879	-35.1269	30.9862	-179.946	-0.562395	91.5384

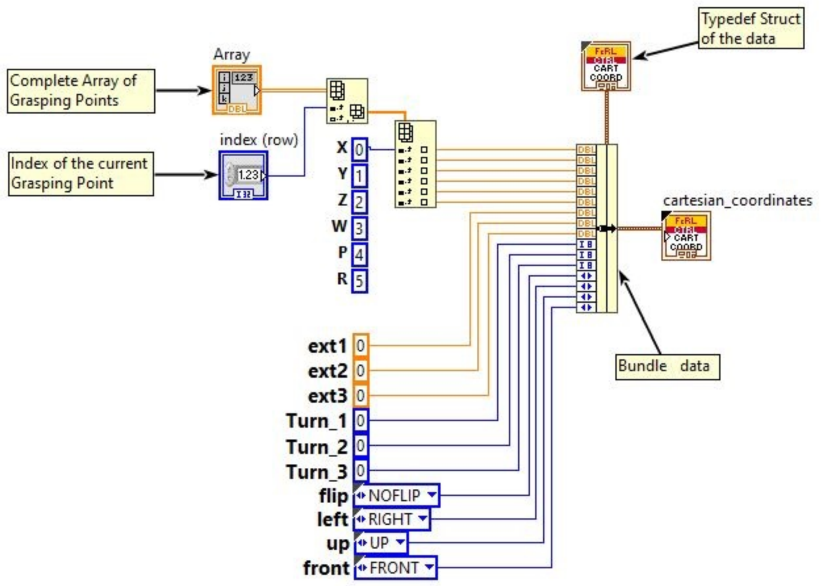
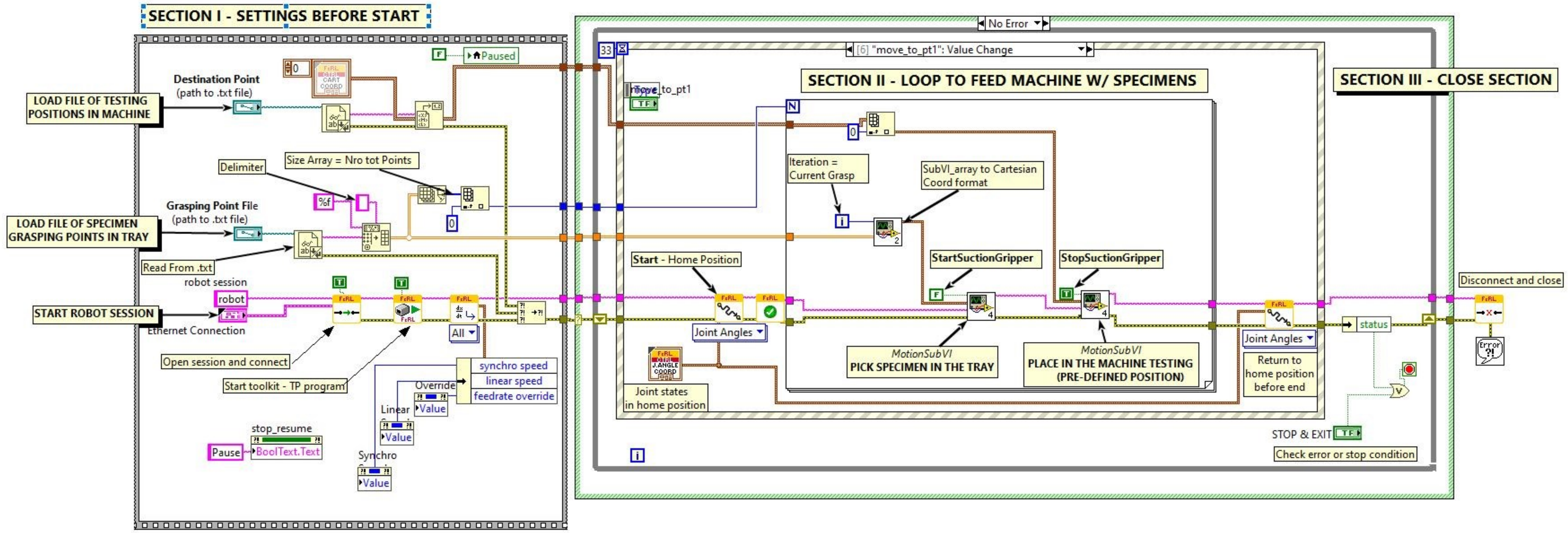


Figure G.1: Block diagram of the array bundle, resulting on Cartesian Coordinates format.

Appendix H

Block diagram of the system


SECTION I - SETTINGS BEFORE START
LOAD FILE OF TESTING POSITIONS IN MACHINE
Destination Point (path to .txt file)
Paused
Delimiter
Size Array = Nro tot Points
%f
LOAD FILE OF SPECIMEN GRASPING POINTS IN TRAY
Grasping Point File (path to .txt file)
Read From .txt
robot session
robot
START ROBOT SESSION
Ethernet Connection
Open session and connect
Start toolkit - TP program
All
synchro speed
linear speed
feedrate override
Override
Linear
Value
Synchro
Value
stop_resume
Pause
BoolText.Text
No Error
[6] "move_to_pt1": Value Change
move_to_pt1
SECTION II - LOOP TO FEED MACHINE W/ SPECIMENS
Iteration = Current Grasp
SubVI_array to Cartesian Coord format
Start - Home Position
StartSuctionGripper
StopSuctionGripper
Joint Angles
MotionSubVI PICK SPECIMEN IN THE TRAY
MotionSubVI PLACE IN THE MACHINE TESTING (PRE-DEFINED POSITION)
Joint states in home position
SECTION III - CLOSE SECTION
Disconnect and close
Joint Angles
Return to home position before end
status
STOP & EXIT
Check error or stop condition