ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
SCUOLA DI INGEGNERIA ED ARCHITETTURA

Department of electrical, electronic and Information engineering "Guglielmo Marconi" - DEI

Master Degree in Automation Engineering

Implementation of a collaborative robot application for closures' quality control

Thesis Advisor:
Prof. Gianluca Palli

Candidate:
Edoardo Guardigli

Co-Advisor:
Dott. Umberto Scarcia

Academic year 2019/2020
Graduation section II

Non è impossibile,
è una sfida.

Contents

List of Figures

Abstract

Among the processes that SACMI IMOLA s.c.r.l has internally, quality control certainly plays a fundamental role.
In particular, Between all the automatic machines that the company produces, the CCM48, a continuous compression moulding machine able, though 48 pistons, to create over 2000 plastic closures / min with max closure diameter 38 mm, is under the lens.
The intent to make the machine competitive on the market leads to the necessity to create an excellent product. This makes quality control of the latter, an aspect of fundamental importance. In this regard, today the closure's control quality procedure is made by two operators that manually, cooperate together. The task is characterized by: LOW frequency, the quality control analysis is realized only once a day on a batch; HIGH repetability, the complete procedure need to be iterate for each cap. Moreover, the operations that are performed by the operators, might be automated through: selection of proper vision sensors for images acquisition, computer vision's algorithms for defects detection and robotic product handling. This is precisely the work that has been carried out in this industrial thesis. The goal is to increase the level of automation in the inspection of the product to highlight possible defects in the process and therefore in the machine. For this purpose, as will be highlighted later, it has been evaluated the possibility to install a collaborative robot to perform this task. The main reason is due to the necessity to do not enormously modify the environment, considering to let the operator works with the robot in a shared workspace.
All the source code used to simulate the environment is available on the personal gitHub account `https://github.com/EdoardoG94/ThesisFolder`.

Introduction

The aim of this thesis is to make a process of crucial importance, such as quality control, semi-autonomous.
Analyzing the characteristics of this process, SACMI IMOLA s.c.r.l. questioned on the feasibility of this approach. For this purpose, the body of the document has been structured in three main chapters:
The first one deals with the study of the problem, analyzing the production's methodologies, advantages and disadvantages that the tecnique means, then the product inspection and all its possible defects. So at the end, the analysis on how, the problem today is addressed.
The second chapter deals first, with some robotics bases and then with the solution of the problem, indicating all the advantages on why this solution turns out to be the best one, focusing on the main norms applicable today and the main monitor technologies.
The third chapter introduce the software used for the simulation, then will be listed all the possible scenarios with the respectively cyclograms for each scenario, looking for the optimal one for the cell design.
Subsequently, a methodology for the representation of the closure will be provided and all the resulting substructures useful for the robotic implementation will be declared.
At the end of the chapter all the results deriving from the simulation and an estimate of the cycle time will be shown.
Finally, in the concluding section, a summary of the work will be expressed by adding some peculiar aspects that could be taken as a starting point for future developments.

Chapter 1

Problem Description

In this chapter will be introduced how a closure is formed and the characteristics that constitute this methodology, diving into the industrial machine used. Then we will touch the closure's control quality, the steps behind the analysis and at the end, how today is managed the process.

1.1 Closure formation

As already said in the introduction, the closures' production is make through CCM48S, so let us go deeper in this process:
CCM is the nomenclature for continuous compression moulding, the first of two metodologies for the creation of the closure's form, in particular, the compression molding is a high-pressure molding process in which the polymer is melted, mixed and homogenized inside a plasticizing unit. A device draws doses of polymer of the exact weight and inserts them into the molds. In order to give a sort if magnitude, the pressure applied to each mold can reach values of about $400Kg/cm^2$. The most valuable feature for this kind of approach are:

- High productivity - due to a shorter cycle time.
- High productivity - due to shorter cycle time.
- Energy savings - due to the lower extrusion temperature.
- Product with better mechanical properties - because plasticizing occurs at low temperatures and with no hit runner, the raw material maintains its characteristics.
- Constant weight and size of the product - due to the lower temperature of the process, a cooler product exiting the moulds means less shrinkage and therefore less size variability.

- Rapid and easy maintenance - thanks to independent moulds, it is possible to replace them quickly and individually.
- Fast and economical colour changeover - the simplicity of the plasticizing unit and the absence of the hot runner speed up the colour changeover and considerably reduce waste of raw material.
  As said, this type of machine is able to realize closures with different diameter, from 28 mm for the classical bottle of water, to 38 mm for the bottle of milk.

The main steps of the process are explained with the following graphical representation:

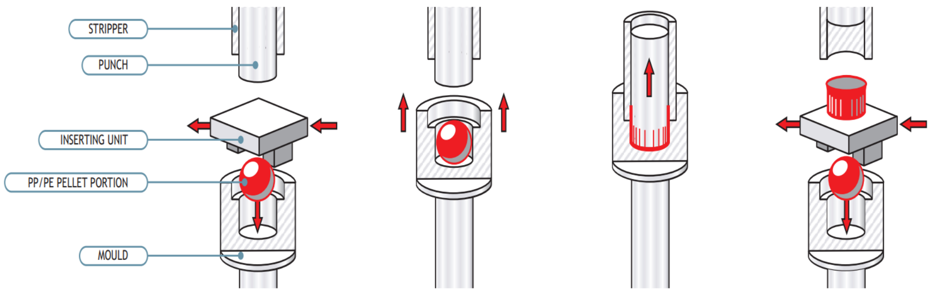


Figure 1.1: Closure's formation process

CCM hydraulic rotary presses are specially designed to produce thermoplastic products by means of compression.
A continuous work cycle is carried out, during which the plastic material is fed by a plasticization unit, cut into suitably sized pallets and then inserted inside the cavities.

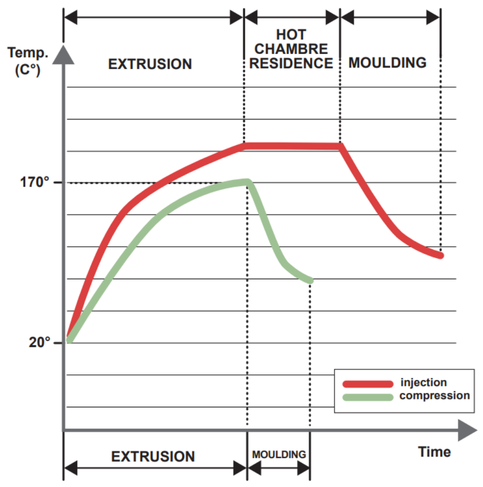


Figure 1.2: Moulding process scheme

A hydraulic system clamps the moulds at the pressure which can be adjusted even while the production cycle is in progress.

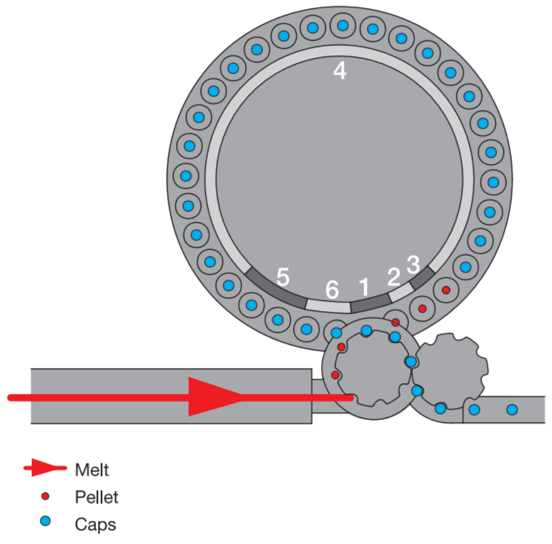


Figure 1.3: CCM hydraulic rotary presses

The task of the extruder is to plasticize the compund, that means melting, mixing and preparing the melt correctly for the lining process. The bulk of energy needed to plasticize the compund comes form the mechanical friction of the screw-barrel-material system.

1.2 Analysis description

Once per day, the products leaving the CCM have to be transported into the analysis department. This procees is made through the following method:

-The department moves a box containing the lot in the laboratory, this box contains 9 laps of the CCM mixed together or 9 boxes containing 1 lap of CCM mixed as well. The control department takes the closures from the box or from the boxes.
Successively, the vision control department uses 2 laps of CCM to perform the analysis: The first lap will be used to verify the quality of the screw and of the plug Seal's external surface, and then it will be used also to misure the imperfection of molding on the top of the plug. The second lap differently, will be used to check more detailed production's errors, that will be explained form the subsection 1.2.4.

1.2.1 Quality of the screw

In order to control perfecly the closure's screw, what the vision department does it to cut the wall of the closure longitudinally in corrispondence of the different sections of the screw, obtaining a sort of flower. Particular attention is put in order to do not cut the internal plug seal. An explicative figure is shown in the following left and the manual operation is shown on the right:

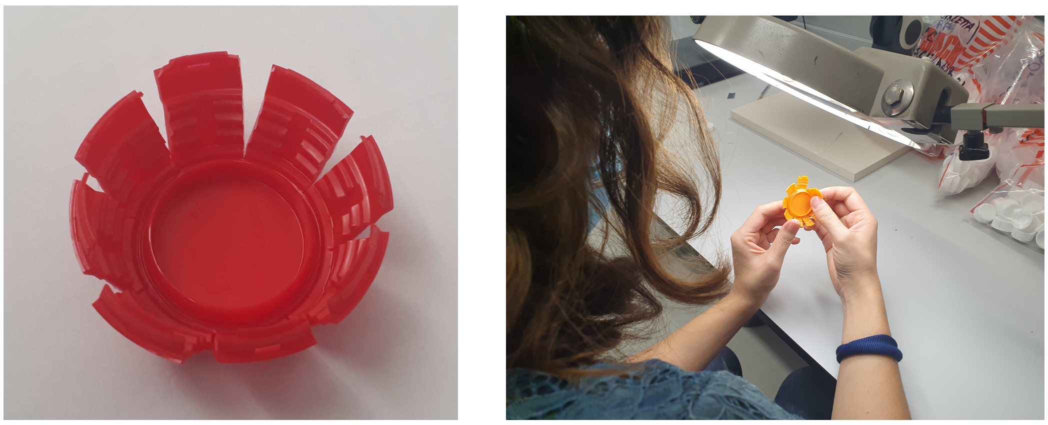


Figure 1.4: Screw Analysis

Proceding in this way is possible to inspect all the screw section in order to detect deviation or deformation from the nominal one:

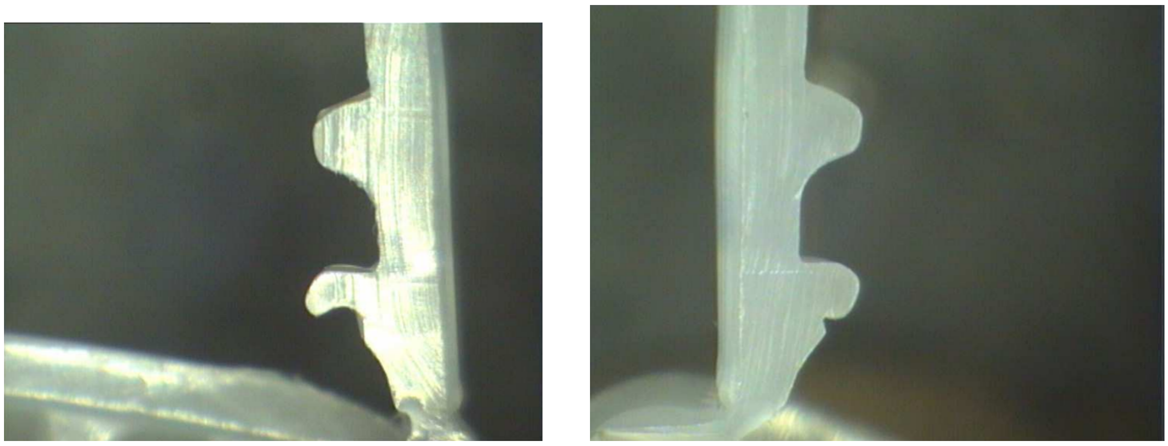


Figure 1.5: Screw Analysis in section

1.2.2 Plug seal external surface

Once the analysis on the screw is conclused, all the petals are removed and the department moves analyzing the ring. Following this approach is possible to detect all the imperfection that could be present, as in the following:

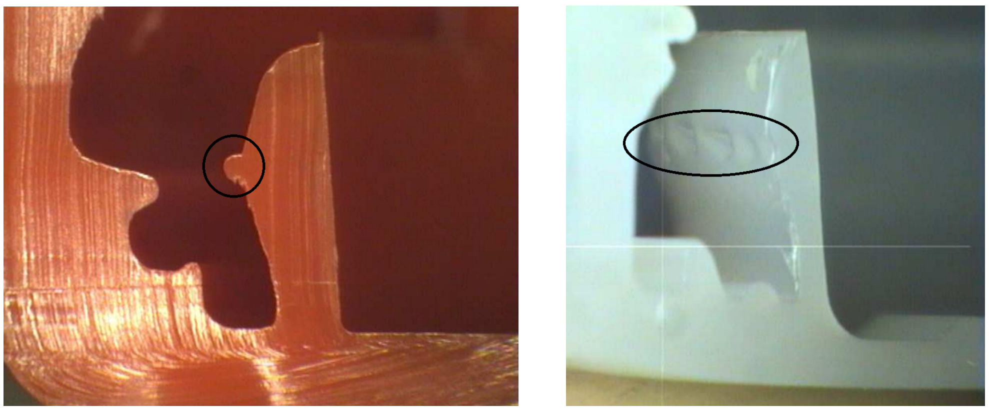


Figure 1.6: Plug seal analysis

As we can clearly see on the left image, there is a deformation towards the outside of the ring, differently from the image on the right which is possible to see a series of holes. All these cases weaken the mechanical structure of the closure no longer causing the optimal isolation.

1.2.3 Plug seal leakage burrs

The last control on the plug seal is performed verifying and misuring of the burrs on the top of the plug:

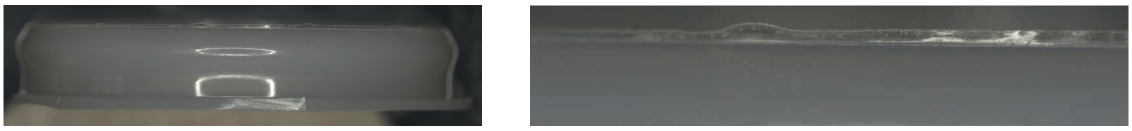
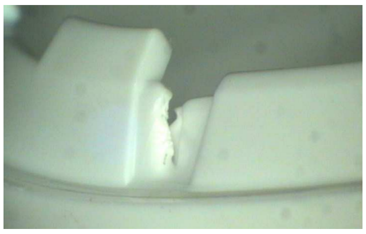


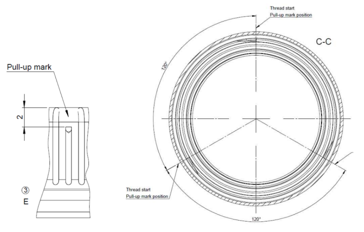
Figure 1.7: Leakage burrs

As it is possible to see, clearer in the right image, this defect is not as crucial as the others plug seal imperfection.

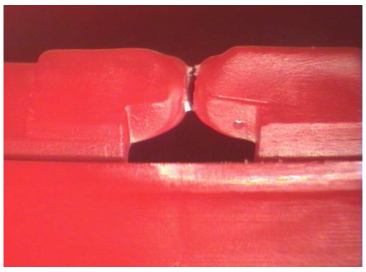
1.2.4 Further imperfections

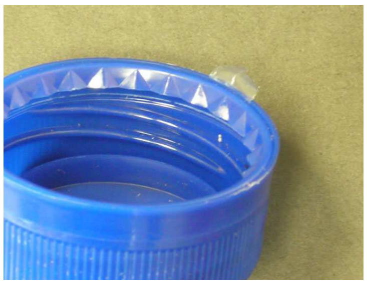
The second lap of the CCM48 is used to analyze all those defects that can be proprietary of the specific model of closure. In the next page I tried to collects all those possible defects that could be identified.


(a) *Fractures on the fin*

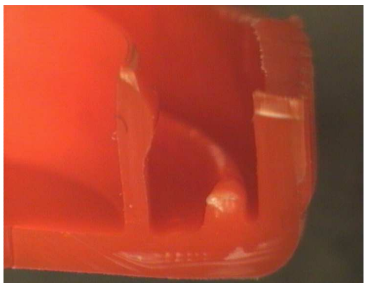


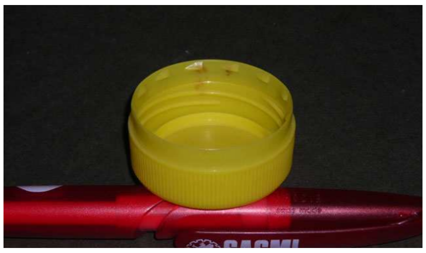
(b) *Control of the PUM position (orientation reference)*

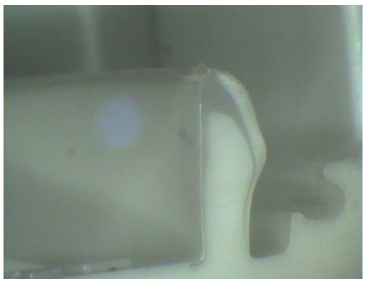

(c) *Uncompleted fin*

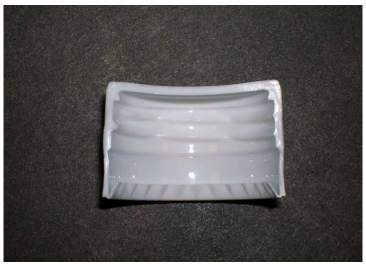

(d) *Leakage burrs on the top of the closure*

(e) *Leakage burrs on the fin and on the top of the fin*


(f) *Top and lateral seal deformations*


(g) *External contaminations from oils*


(h) *Not optimal colourant distribution*


(i) *Excessive convessity of the capsule's base*

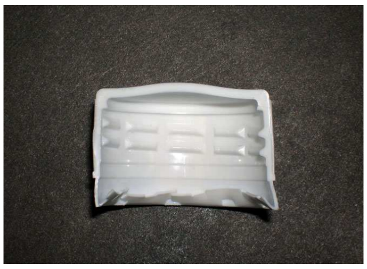

(j) *Excessive concavity of the capsule's base*

Figure 1.8: Second lap possible imperfections

Today's procedure

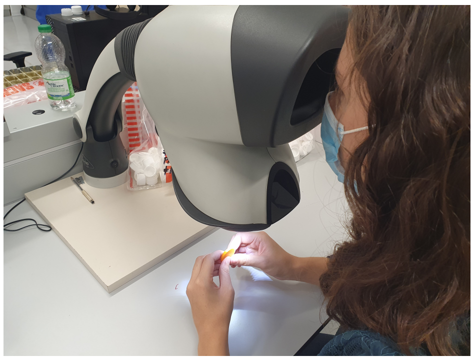
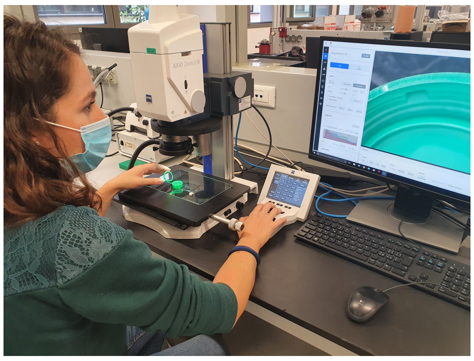



Figure 1.9: Manual operations

As it is clearly visible from the two images above, the manual process that today is perfromed, is characterized by an high level of accuracy and an high level of experience. These two main properties would require the evolution through a solution that combines both factors. Moreover the inspection time spent to complete the entire process is 300 minutes.

Chapter 2

Robotic and collaborative robotic notions

The necessity to keep a certain grade of manual skill combined with the desire of automatize the process, requires a solution to the problem that merge both worlds.
Therefore it is for this reason that one of the solutions that best lends itself to being in line with this thought is the adoption of a collaborative robot.

Without focusing on what a collaborative robot actually is now, there is the necessity first to define some robotic's basics. So as mentioned in the introduction, this chapter deals with the most suitable solution, mentioning some robotic bases, passing through collaborative robotics and in particular touching the norms applicable today.

2.1 Robotic's introduction

The image of the robot as a mechanical artifact starts in the 1940s when the Russian Isaac Asimov, the well-known science fiction writer, conceived the robot as an automaton of human appearance but devoid of feelings. Its behaviour was dictated by a “positronic” brain programmed by a human being in such a way as to satisfy certain rules of ethical conduct. The term robotics was then introduced by Asimov as the science devoted to the study of robots which was based on the three fundamental laws:

- A robot may not injure a human being or, through inaction, allow a human being to come to harm.
- A robot must obey the orders given by human beings, except when such orders would conflict with the first law.
- A robot must protect its own existence, as long as such protection does not conflict with the first or second law.

These laws established rules of behaviour to consider as specifications for the design of a robot, which since then has attained the connotation of an industrial product designed by engineers or specialized technicians. According to a scientific interpretation of the science-fiction scenario, the robot is seen as a machine that, independently of its exterior, is able to modify the environment in which it operates. With reference to this definition, a robotic system is in reality a complex system, functionally represented by multiple subsystems. The essential component of a robot is the mechanical system endowed, in general, with a locomotion apparatus (wheels, crawlers, mechanical legs) and a manipulation apparatus (mechanical arms, end-effectors, artificial hands). Both locomotion and manipulation, is provided by an actuation system which animates the mechanical components of the robot. The concept of such a system refers to the context of motion control, dealing with servomotors, drives and transmissions. The capability for perception is entrusted to a sensory system which can acquire data on the internal status of the mechanical system (such as position transducers) as well as on the external status of the environment (such as force sensors and cameras).

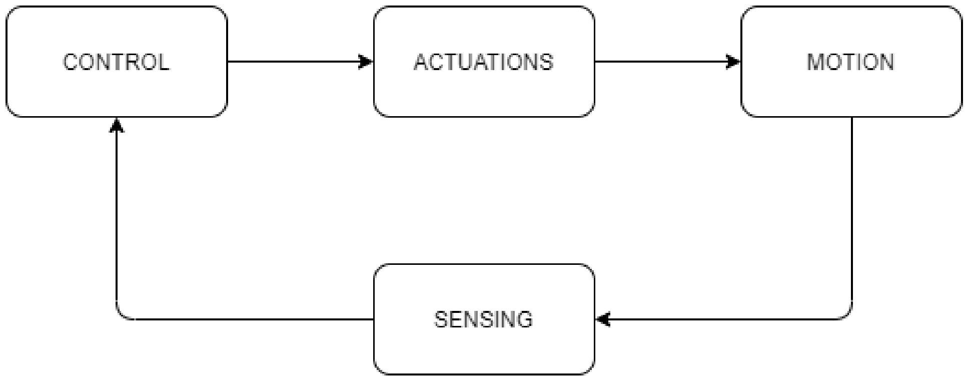


Figure 2.1: Subgroup decomposition

The key feature of a robot is its mechanical structure. Robots can be classified as those with a fixed base, robot manipulators, and those with a mobile base, mobile robots.

2.1.1 Degree of freedom

A manipulator may be mechanically described as the interconnection of rigid bodies (links) through kinematic pairs (joints).A joint is an element that constrains one or more relative motion directions between two rigid bodies (links). In the most general case, joints are of two types:

- Rotoidal: the motion is a rotation about a fixed axis on the left of figure 2.2.

- Prismatic: There is a traslation along a fixed axis on the right of figure 2.2.

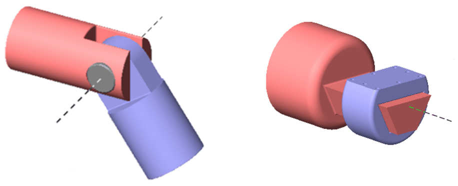


Figure 2.2: Rotoidal and prismatic joint

In an open kinematic chain, each prismatic or revolute joint provides the structure with a single degree of freedom (DOF). A prismatic joint creates a relative translational motion between the two links, whereas a revolute joint creates a relative rotational motion between the two links. Revolute joints are usually preferred to prismatic joints in view of their compactness and reliability. On the other hand, in a closed kinematic chain, the number of DOFs is less than the number of joints in view of the constraints imposed by the loop. The degrees of freedom should be properly distributed along the mechanical structure in order to have a sufficient number to execute a given task. In the most general case of a task consisting of arbitrarily positioning and orienting an object in three-dimensional (3D) space, six DOFs are required, three for positioning a point on the object and three for orienting the object with respect to a reference coordinate frame. If more DOFs than task variables are available, the manipulator is said to be redundant from a kinematic viewpoint. If a joint has k degrees of freedom, then the relative configuration between two rigid bodies may be expressed as a function of k variables $q_1, q_2, ..., q_k$, called joint variables, in which it is possible to consider q as:

- the amplitude of rotation (θ) for the rotational joint
- the amplitude of tralastion (d) for the prismatic joint.

Considering a serial kinematic chain, in which we have $n + 1$ links $(L_0, L_1, ..., L_n)$ interconnected by n joints $(J_0, J_1, ..., J_n)$, and denoting by K_i the degrees of freedom of the i_{th} joint, with $i = 1, ...n$ Then the manipulator configuration depends on N_{dof} independent variables, where:

$$N_{dof} = \sum_{i=1}^{n} k_i$$

N_{dof} usually represents the number of actuators and is equal to n for a manipulator with n + 1 links and rotoidal/prismatic joints.

2.1.2 Kinematic model

A manipulator can be schematically represented from a mechanical viewpoint as a kinematic chain of rigid bodies (links) connected by means of revolute or prismatic joints. One end of the chain is constrained to a base, while an end-effector is mounted to the other end. The resulting motion of the structure is obtained by composition of the elementary motions of each link with respect to the previous one. Therefore, in order to manipulate an object in space, it is necessary to describe the end-effector position and orientation. This section focuses on the derivation of the direct kinematics equation through a systematic, general approach based on linear algebra. This allows the end-effector position and orientation (pose) to be expressed as a function of the joint variables of the mechanical structure with respect to a reference. frame

Rigid body frame

A rigid body is completely described in space by its position and orientation with respect to a reference frame, considering $O - xyz$ be the orthogolan reference frame and x, y, z be the unit vectors of the frame axes.

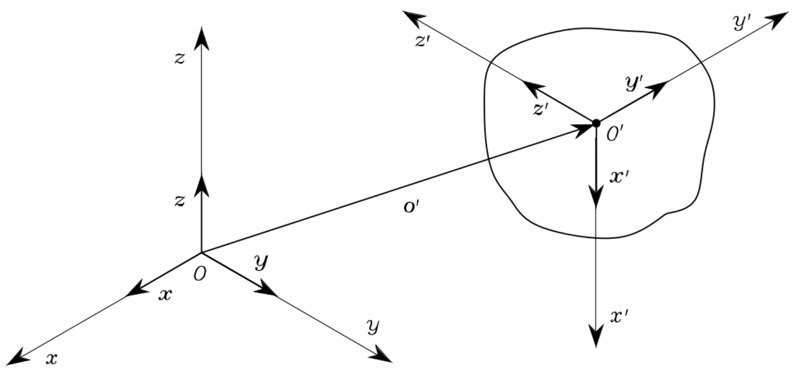


Figure 2.3: Rigid body frame

The position of a point O' on the rigid body with respect to the coordinate frame O–xyz is expressed by the relation:

$$o' = o'_x x + o'_y y + o'_z z$$

Whew o'_x, o'_y, o'_z denote the component of the vector $o' \in \mathbb{R}^3$ along the frame axes; teh position of O' can be compactly written as $(3x1)$ vector:

$$\begin{bmatrix} o'_x \\ o'_y \\ o'_z \end{bmatrix}$$

In order to describe the rigid body orientation, it is convenient to consider an orthonormal frame attached to the body and express its unit vectors with respect to the reference frame. So considering $O' - x'y'z'$ be such a frame with origin in O' and considering x', y', z' be the vecrors of the frame axes, then these vectors are expressed with respect to the reference frame $O - xyz$ by the following system of equations:

$$\begin{aligned} x' &= x'_x x + x'_y y + x'_z z \\ x' &= y'_x x + y'_y y + y'_z z \\ x' &= z'_x x + z'_y y + z'_z z \end{aligned}$$

So finally we have the direction of each vector are the direction cosine of the axes of the frame $O' - x'y'z'$ with respect to the reference frame $O - xyz$.

Rotation and Homogeneous transformation matrices

In matrix form, the system composed of three equation above, can be written as:

$$R = \begin{bmatrix} o'_x & o'_y & o'_z \end{bmatrix} = \begin{bmatrix} x'_x & y'_x & z'_x \\ x'_y & y'_y & z'_y \\ x'_z & y'_z & z'_z \end{bmatrix}$$

This matrix can be called as Rotation matrix.
It is worth noting that the Rotation matrix R is an orthogonal matrix since each coloumn vector of R are mutually orthogonal each other considering that they represent the unit vectors of an orthogonal frame and we can notice that they have also unit norm:

$$\begin{aligned} x'^T y' = 0, y'^T z' = 0, z'^T x' = 0 \\ x'^T x' = 1, y'^T y' = 1, z'^T z' = 1 \end{aligned}$$

The property of orthogonality means that:

$$R^T R = I_3, \text{ where } I_3 \text{ is the } 3 \times 3 \text{ identity matrix}$$

The matrix previously define belongs to the special orthonormal group SO(m) of real $(m \times m)$ matrices with orthonormal coloumns and determinant equal to 1 if the frame is $right - handed$, inversely if the frame is $left - handed$ the $detR = -1$.

Considering the following frame:

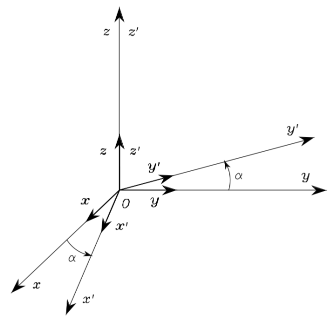


Figure 2.4: Rotation of frame $O - xyz$ about z

This frame is a frame that can be obtained via elementary rotations of the reference one about of the coordinate axes. Suppose then that the reference frame $O - xyz$ is rotated by and angle α about axis z, and so consider $O - x'y'z'$ be the rotated frame. The unit vectors of the new frame can be described in terms of their components with respect to the reference frame.
Considering the frames that can be obtained via elementary rotations of the reference frame about one of the coordinate axes, these rotations are positive if they are made $counter - clockwise$ about the relative axis. The unit vectors of the new frame can be described in terms of their components with respect to the reference frame:

$$x' = \begin{bmatrix} cos\alpha \\ sin\alpha \\ 0 \end{bmatrix} \; y' = \begin{bmatrix} -sin\alpha \\ cos\alpha \\ 0 \end{bmatrix} \; z' = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

So grouping each vector we obtain:

$$R_z(\alpha) = \begin{bmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The same can be made considering the rotation about the $y - axis$ and $x - axis$, though the angles β and γ respectively, obtaining:

$$R_y(\beta) = \begin{bmatrix} cos\beta & 0 & sin\beta \\ 0 & 1 & 0 \\ -sin\beta & 0 & cos\beta \end{bmatrix} \; R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\gamma & -sin\gamma \\ 0 & sin\gamma & cos\gamma \end{bmatrix}$$

Consider the following figure:

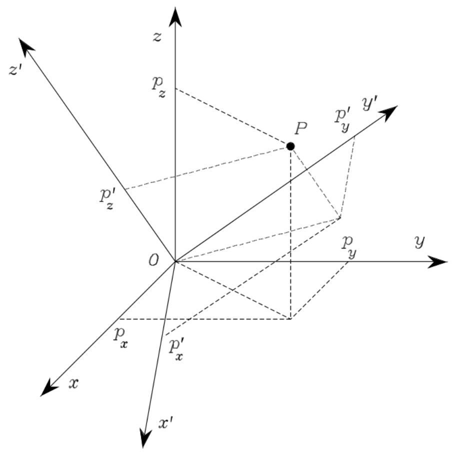


Figure 2.5: Representation of a point P using different coordinate frames

In order to fully understand the geometrical meaning of the rotation matrix and of the homogeneous transformation matrix, it is necessary to understand the different representation of the same vector using different bases.

So, the point P can be represented with respect to frame $O - xyz$:

$$p = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

and can be represented with respect to the frame $O - x'y'z'$ as:

$$p' = \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix}$$

Taking account now the 2-D case, it is the possible to rotate a vector with respect to an axis using the rotation matrix, i.e., consider the following figure:

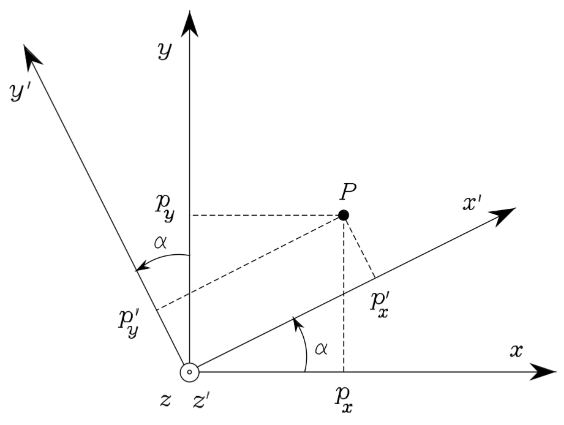


Figure 2.6: Representation of a point P in two different planar frames

A rotation matrix can be also interpreted as the matrix operator allowing rotation of a vector by a given angle about an arbitrary axis in space. Indeed, let p' be a vector in the reference frame O–$x'y'z'$; in view of orthogonality of the matrix R, the product $R\ p'$ yields a vector p with the same norm as that of p' but rotated with respect to p' according to the matrix R. So since the case in exam is a planar rotation with respect to the third axis coming out from the paper, it is easy to recognize that p can be expressed as, rememebring that $R_z(\alpha)$ is the rotation matrix belong z axis:

$$p = R_z(\alpha)p'$$

Rotation matrices give a redundant description of frame orientation; indeed, they are characterized by nine elements which are not independent but, thanks to the orthogonality conditions, related by six constraints. This implies that three parameters are sufficient to describe the orientation of a rigid body in space. A minimal representation of orientation can be obtained by using a set of three angles $\varphi = [\phi\ \vartheta\ \psi]^T$. Consider the rotation matrix expressing the elementary rotation about one of the coordinate axes as a function of a single angle.
Then, a generic rotation matrix can be obtained by composing a suitable sequence of three elementary rotations while guaranteeing that two successive rotations are not made about parallel axes. This implies that 12 $(3 \times 2 \times 2)$distinct sets of angles are allowed out of all 27 possible combinations $(3 \times 3 \times 3)$.

The following combination, gives an example of ZYZ rotation using elementary rotation matrices:

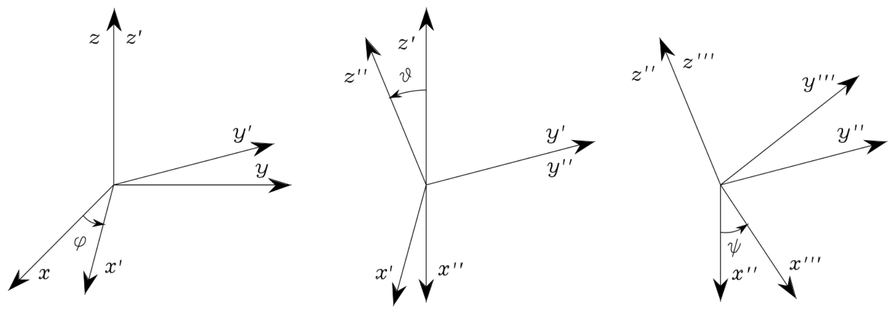


Figure 2.7: ZYZ elementary rotation

In the most general case, the position of a rigid body in space is expressed in terms of the position of a suitable point on the body with respect to a reference frame (translation), while its orientation is expressed in terms of the components of the unit vectors of a frame attached to the body with respect to the same reference frame (rotation). Taking into account of the following figure, and consider an arbitrary point P.

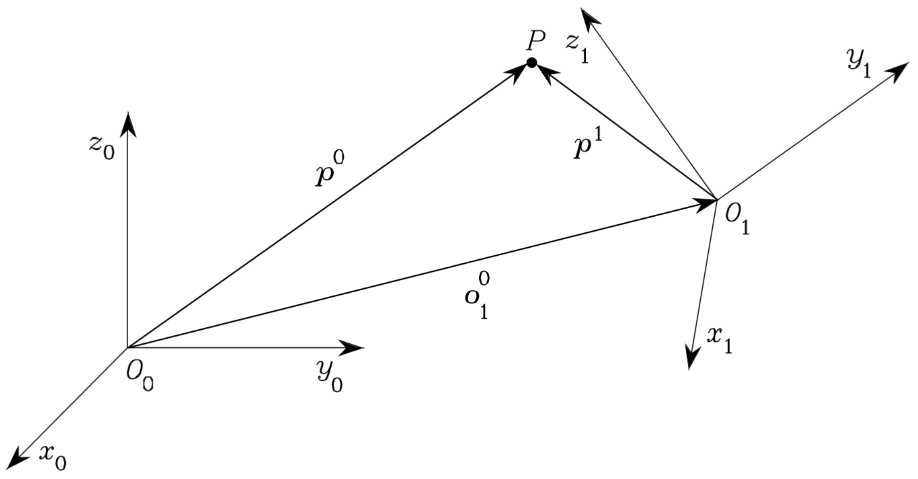


Figure 2.8: Point P form different frames

Let now p^0 be the vector of coordinates of P with respect to the frame $O_0 - x_0y_0z_0$. Consider now another frame in space $O_1 - x_1y_1z_1$.Let o_1^0 be the vector describing the origin of Frame 1 with respect to Frame 0, and R_1^0 be the rotation matrix of Frame 1 with respect to Frame 0. Let also p^1 be the vector of coordinates of P with respect to Frame 1. On the basis of simple geometry, the position of point P with respect to the reference frame can be expressed as:

$$p^0 = o_1^0 + R_1^0 p^1$$

This equation represents the coordinate transformation ($\textit{transaltion} + \textit{rotation}$) of a bound vector between two different frames. The transformation can also be inverted premultiplying both side by R_1^{0T}, obtaining:

$$P^1 = -R_1^{0T} o_1^0 + R_1^{0T} p^0$$

that can be rewritten as:

$$P^1 = -R_0^1 o_1^0 + R_0^1 p^0$$

As before in order to achieve a compact representation of the relationship between the coordinates of the same point in two different frames, the homogeneous representation of a generic vector p can be introduced as the vector $\tilde{p}$ formed by adding a fourth unit component:

$$\tilde{p} = \begin{bmatrix} p \\ 1 \end{bmatrix}$$

Following this notation it is possible to express p^0 and p^1 previously defined, using the following 4×4 matrix:

$$A_1^0 = \begin{bmatrix} R_1^0 & o_1^0 \\ \\ o^T & 1 \end{bmatrix}$$

which, accordingly to $\tilde{p}$ is called homogeneous transformation matrix. As can be easily seen, the transformation of a vector from Frame 1 to Frame 0 is expressed by a single matrix containing the rotation matrix of Frame 1 with respect to Frame 0 and the translation vector from the origin of Frame 0 to the origin of Frame 1. So the coordinate transformation can be rewritten in a the compact form as:

$$\tilde{p}^1 = A_0^1 \tilde{p}^0 = (A_1^0)^{-1} \tilde{p}^0$$

It is fundamental to notice that in this case, differently from the rotation matrix, the homogeneous transformation matrix is no longer orthogonal, hence in general:

$$A^{-1} \neq A^T.$$

Summarizing, the homogeneous transformation matrix is able to express the coordinate transformation between two frames in compact form.

Forward kinematic

As already said, a manipulator consists of a series of rigid bodies (links) connected by means of kinematic pairs or joints. Joints can be essentially of two types: revolute and prismatic. The whole structure forms a kinematic chain. One end of the chain is constrained to a base (for *non-mobile robotics*). An end-effector (which can be a gripper or more in general, a tool) is connected to the other end allowing manipulation of objects in space. The mechanical structure of a manipulator is characterized by a number of degrees of freedom (DOFs) which uniquely determine its posture. Each DOF is typically associated with a joint articulation and constitutes a joint variable. The aim of direct kinematics is to compute the pose of the end-effector as function of the joint variables. Taking as reference the following figure:

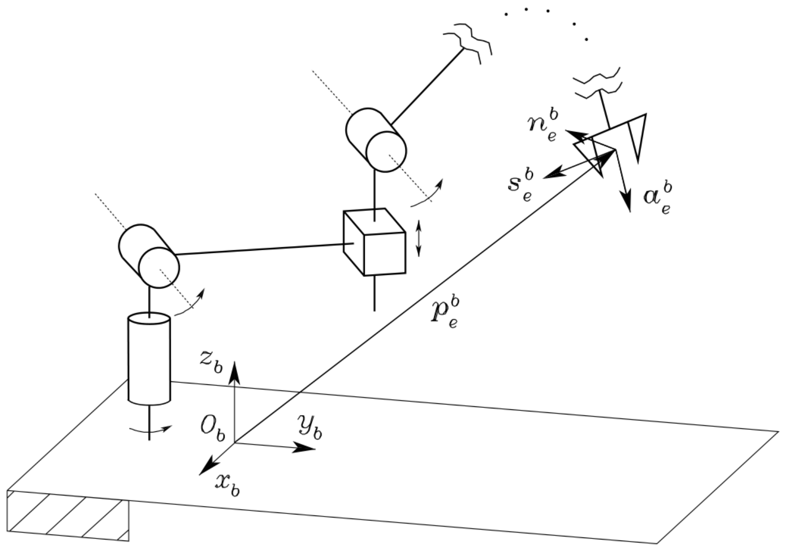


Figure 2.9: Position and orientation of the end-effector with respect to the base

It has been previously illustrated that the pose of a body with respect to a reference frame is described by the position vector of the origin and the unit vectors of a frame attached to the body. So considering the homogeneous transformation matrix, the direct kinematics function expressed with respect to a reference frame $O_b - x_b y_b z_b$ is :

$$T_e^b(q) = \begin{bmatrix} n_e^b(q) & s_e^b(q) & a_e^b(q) & p_e^b(q) \\ \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where q is the $(n \times 1)$ vector of joint variables, n_e, s_e, a_e are the unit vectors of a frame attached to the end-effector and p_e is the position vector of the origin of such a frame with respect to the origin of the base frame $O_b - x_b y_b z_b$. It is important to note that, n_e, s_e, a_e and p_e are function of the joint variables q. Regarding the frames, the frame $O_b - x_b y_b z_b$, is termed base frame, inversely, the frame attached to the end-effector is called end-effector frame and is conveniently chosen according to the task. If, for instance, the end-effector is a gripper, the origin of the end-effector frame is located at the centre of the gripper, the unit vector a_e is chosen in the approach direction to the object, the unit vector s_e is chosen normal to a_e in the sliding plane of the jaws, and the unit vector n_e is chosen normal to the other two so that the frame (n_e, s_e, a_e) is right-handed. Before to go on, could be useful to make an example of what just explained. Consider the following chain:

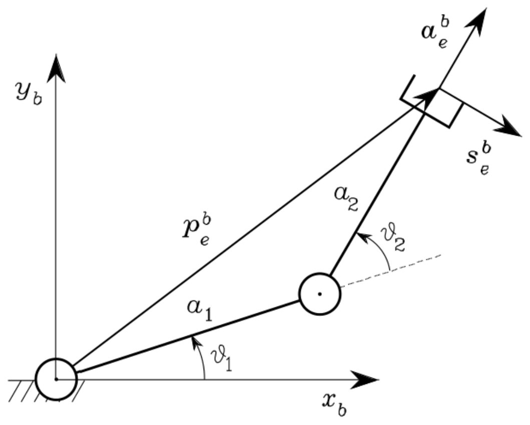


Figure 2.10: 2-D two link planar arm

Taking into account the theory previously said, it is possible to define the homogeneous transformation matrix of the end effector with respect to the base as:

$$T_e^b(q) = \begin{bmatrix} n_e^b(q) & s_e^b(q) & a_e^b(q) & p_e^b(q) \\ \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & s_{12} & c_{12} & a_1c_1 + a_2c_{12} \\ 0 & -c_{12} & s_{12} & a_1s_1 + a_2s_{12} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In this example, the armw was composed of just two links, so it is easy to compute $p_e^b(q)$, but, what if the arm is composed of more than just two links? Consider then the following chain:

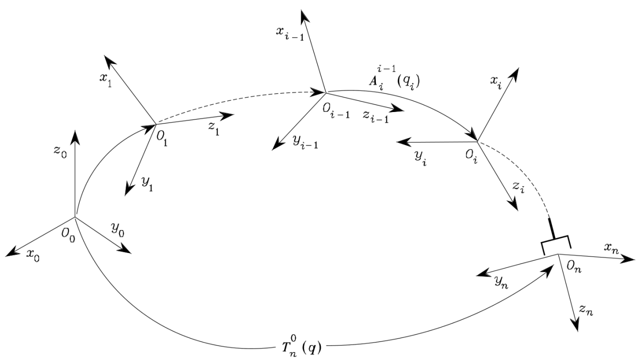


Figure 2.11: Homogeneous transformation matrix in an open kinematic chain

The chain under exam is costituted by $n+1$ links connected by n joints, where Link 0 is conventionally fixed to the ground. It is assumed that each joint provides the mechanical structure with a single DOF, corresponding to the joint variable. The construction of an operating procedure for the computation of direct kinematics is naturally derived from the typical open kinematic chain of the manipulator structure. In fact, since each joint connects two consecutive links, it is reasonable to consider first the description of kinematic relationship between consecutive links and then to obtain the overall description of manipulator kinematics in a recursive fashion. So defining a coordinate attached to each link, from link 0 to link n, Then the coordinate transformation describing the position and orientation of frame n with respect to frame 0 is given by:

$$T_n^0(q) = A_1^0(q_1)A_2^1(q_2)...A_n^{n-1}(q_n)$$

As requested, the computation of the direct kinematics function is recursiv and is obtained in a systematic manner by simple products of the homogeneous transformation matrices $A_i^{i-1}(q_i)(for\ i = 1, ..., n)$, each of which is function of a single joint variable. With reference to the base and end-effector frame defined before, we have:

$$T_e^b = T_0^b T_n^0(q) T_e^n$$

Where $T_e^b(q)$ and T_e^n are two (typically) constant homogeneous transformations describing the position and orientation of Frame 0 with respect to the base frame, and of the end-effector frame with respect to Frame n.

In order to compute the direct kinematic equation for an open-chain manipulator according to the recursive expression, a systematic,method has to be derived to define the relative position and orientation of two consecutive links, moving the problem to determine two frames attached to the two links and compute the coordinate transformations between them. In general, the frames can be arbitrarily chosen as long as they are attached to the link they are referred to. Nevertheless, it is convenient to set some rules also for the definition of the link frames. Considering the following figure, let Axis i denote the axis of the joint connecting Link $i-1$ to Link i; the so-called Denavit–Hartenberg convention (DH) is adopted to define link Frame i:

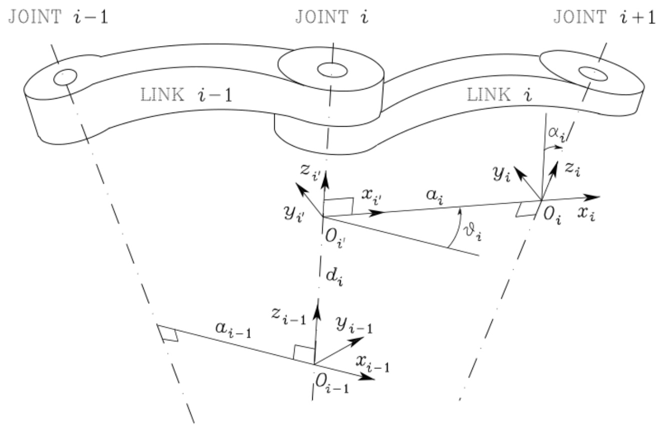


Figure 2.12: Denavit-Hartenberger notation

- Choose axis z_i along the axis of joint $i+1$
- Locate the origin $=_i$ at the intersection of axis Z_i with the common normal to axes Z_{i-1} and axis Z_i. Also, locate $O_{i'}$ at the intersection of the commoon normal with the axis z_{i-1}.
- Choose axis x_i along the common normal to axes z_{i-1} and z_i with direction from joint i to joint $i-1$.
- Choose axis y_i so as to complete a right-handed frame.

The Denavit–Hartenberg convention gives a nonunique definition of the link frame in the following cases:

- For Frame 0, only the direction of axis z_0 is specified; then O_0 and x_0 can be arbitrarily chosen.
- For Frame n, since there is no Joint $n+1$, z_n is not uniquely defined while x_n has to be normal to axis z_{n-1}. Typically, Joint n is revolute, and thus z_n is to be aligned with the direction of z_{n-1}.
- When two consecutive axes are parallel, the common normal between them is not uniquely defined.
- When two consecutive axes intersect, the direction of x_i is arbitrary.

- When Joint i is prismatic, the direction of z_{i-1} is arbitrary.

Nevertheless, in all such cases, the indeterminacy can be exploited to simplify the procedure, for instance, considering the axes of two consecutive frames as pararrel. Then, once the link frames, have been established, it is possible, using the following parameters, to define the position and orientation of frame i with respect to frame $i-1$:

- a_i distance between O_i and $O_{i'}$.
- d_i coordinate of $O_{i'}$ along z_{i-1}.
- α_i angle between axes z_{i-1} and z_i about axis x_i to be taken positive when the rotation is concordant with the right-hand law.
- θ_i angle between axes x_{i-1} and x_i about axis z_{i-1} to be taken positive when the rotation is concordant with the right-hand law.

The first and the third parameters are always constant and depend only on the geometry of connection between consecutive joints established by link i.
For the other two parameters, only one is variable depending on the type of joint that connects Link $i-1$ to Link i. In particular:

- If joint i is revolute, then the variable is θ_i.
- If joint i is prismatic, then the variable is d_i.

Finally, it is possible to express the coordinate transformation between frame i and frame $i-1$ through the following matrix:

$$A_i^{i-1}(q_i) = A_{i'}^{i-1} A_i^{i'} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

To summarize, the Denavit–Hartenberg convention allows the construction of the direct kinematics function by composition of the individual coordinate transformations expressed by the previous define matrix into one homogeneous transformation matrix.

2.1.3 Joint and work space

The direct kinematics equation of a manipulator allows the position and orientation of the end effector frame to be expressed as a function of the joint variables with respect to the base frame in general. If a task is to be specified for the end-effector, it is necessary to assign the end-effector position and orientation, eventually as a function of time (trajectory). This is quite easy for the position. On the other hand, specifying the orientation through the unit vector triplet (n_e, s_e, a_e) is quite difficult, since their nine components must be guaranteed to satisfy the orthonormality constraints imposed at each time instant. The problem of describing end-effector orientation admits a natural solution ,in particular, in this case, indeed, a motion trajectory can be assigned to the set of angles chosen to represent orientation. Therefore, the position can be given by a minimal number of coordinates with regard to the geometry of the structure, and the orientation can be specified in terms of a minimal representation (Euler angles) describing the rotation of the end-effector frame with respect to the base frame. In this way, it is possible to describe the end-effector pose by means of the $(m \times 1)$ vector, with $m \leq n$:

$$x_e = \begin{bmatrix} p_e \\ \phi_e \end{bmatrix}$$

where p_e describes the end-effector position and ϕ_e its orientation.

This representation of position and orientation allows the description of an end-effector task in terms of a number of inherently independent parameters.
The vector x_e is defined in the space in which the manipulator task is specified, defining a space, tipycally called operational space. On the other hand, the joint space (configuration space) denotes the space in which the $(n \times 1)$ vector of joint variables is defined:

$$q = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$$

where, as before, $q_i = \theta_i$ for a revolute joint and $q_i = d_i$ for a prismatic joint. Taking into account the dependence of position and orientation from the joint variables, the direct kinematics equation can be written also as:

$$x_e = k(q)$$

The $(m \times 1)$ vector function $k(\cdot)$ allows computation of the operational space variables from the knowledge of the joint space variables. It is interesting to notice that the dependence of the orientation components of the function $k(q)$ on the joint variable is not easy to express except fot simple cases. Indeed, the most general case of six-dimensional operational space ($m = 6$), the computation of three components of the function $\phi_e(q)$

cannot be performed in closed form, but goes through the computation of the elements of the rotation matrix: $n_e(q)$, $s_e(q)$, $a_e(q)$.

Taking as reference the operational space, an index of robot performance is the so-called workspace; this is the region described by the origin of the end-effector frame when all the manipulator joints execute all possible motions.Under normal condition it is customary to distinguish between reachable workspace and dexterous workspace. The latter is the region that the origin of the end-effector frame can describe while attaining different orientations, while the former is the region that the origin of the end-effector frame can reach with at least one orientation. It is easy to understand that the daxterous workspace is a subspace of the reachable workspace, indeed, a manipulator witt less than six DOFs cannot take any arbitrary position and orientation in space. The workspace is characterized by the manipulator geometry and the mechanical joint limits. For an n-DOF manipulator, the reachable workspace isthe geometric locus of the points that can be achieved by considering the direct kinematics equation:

$$p_e = p_e(q) \qquad q_{im} \leq q_i \leq q_{iM} \; i = 1, ..., n$$

where $q_{im}(q_{iM})$ denotes the minimum(maximum) limit at the joint i. This volume is finite, closed, connected and thus is defined by its bordering surface. Normally, the manipulator workspace (without end-effector) is reported in the data sheet given by the robot manufacturer in terms of a top view and a side view. It is necessary since it helps to evaluate robot performance for a desired application.

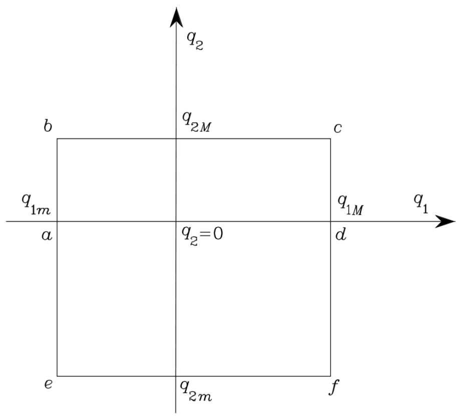


Figure 2.13: Two-link arm admissible configuration region

With the previous and following draws, can be clarly understandable, what a joint space and a work space are graphically, and what it means having a finite, closed and connected volume.

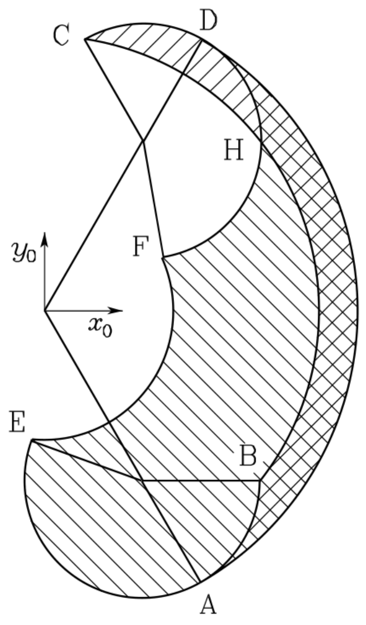


Figure 2.14: Two-link arm work space

In a real manipulator, for a given set of joint variables, the actual values of the operational space variables deviate from those computed via direct kinematics. The direct kinematics equation has indeed a dependence from the Denavit-Hartenberg parameters . If the mechanical dimensions of the structure differ from the corresponding parameter of the table because of mechanical tolerances, a deviation arises between the position reached in the assigned posture and the position computed via direct kinematics. Such a deviation is defined accuracy; this parameter attains typical values below one millimeter and depends on the structure as well as on manipulator dimensions. Accuracy varies with the end-effector position in the workspace and it is a relevant parameter.
Another parameter that is usually listed in the performance data sheet of an industrial robot is repeatability, which is a measure of the manipulator's ability to return to a previously reached position Repeatability depends not only on the characteristics of the mechanical structure but also on the transducers and controller; it is expressed in metric units and is typically smaller than accuracy.

2.1.4 Inverse kinematic

The direct kinematics equation, establishes the functional relationship between the joint variables and the end-effector position and orientation. On the other side, the inverse kinematics problem consists of the determination of the joint variables corresponding to a given end-effector position and orientation. The solution to this problem is of crucial portance in order to transform the motion specifications, assigned to the end-effector in the work space, into the corresponding joint space motions that allow execution of the desired motion or task.
Differently from the direct kinematics equation in which the end-effector position and rotation matrix are computed in a unique manner, once the joint variables are known, the inverse kinematics problem is much more complex fo the following reason:

- The equations to solve are in general nonlinear, and thus it is not always possible to find a closed-form solution.
- Multiple solutions may exist.
- Infinite solutions may exist, e.g., in the case of a kinematically redundant manipulator.
- There might be no admissible solutions, in view of the manipulator kinematic structure

Recalling the daxterous workspace, the existence of solutions is guaranteed only if the given end-effector position and orientation belong to the manipulator dexterous workspace.
On the other hand, the problem of multiple solutions depends not only on the number of DOFs but also on the number of non-null DH parameters; generally, the greater the number of non-null parameters, the greater the number of admissible solutions. Taking as example a six-DOF manipulator without mechanical joint limits, there are in general up to 16 admissible solutions. Such occurrence demands some criterion to choose among admissible solutions. The existence of mechanical joint limits may eventually reduce the number of admissible multiple solutions for the real structure. Computation of closed-form solutions requires either algebraic intuition to find those significant equations containing the unknowns or on the other side, geometric intuition in order to find those significant points on the structure that allows expressing position and/or orientation as a function of a reduced number of unknowns.

As for the direct kinematic approach, in order to clarify the ideas, it could be of help, make an example on three-link planar arm, so consider the following, manipulator:

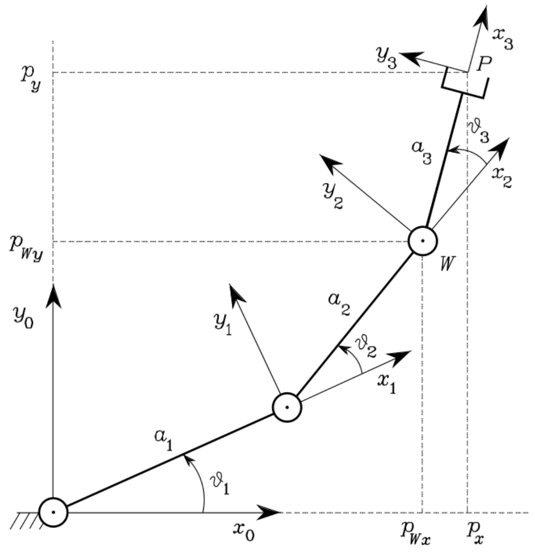


Figure 2.15: Three-link arm

As already pointed out, it is convenient to specify position and orientation in terms of a minimal number of parameters: the two coordinates p_x, p_y and the angle ϕ with axis x_0, in this case.
A first algebric solution technique could be:

$$\phi = \theta_1 + \theta_2 + \theta_3$$

in which:

$$\begin{aligned} p_{Wx} &= p_x - a_3 c_\phi = a_1 c_1 + a_2 c_{12} \\ p_{Wy} &= p_y - a_3 s_\phi = a_1 s_1 + a_2 s_{12} \end{aligned}$$

Combining the 2 equations it is possible to remove the dependency from the angle θ_1, allowing to obtain c_2 and s_2. Once c_2 and s_2 are known, it is possible to figure out θ_2 through:

$$\theta_2 = Atan2(s_2, c_2)$$

The same can be done with, θ_1 in which, once θ_2 is known. Finally the angle θ_3 can be found from:

$$\theta_3 = \phi - \theta_1 - \theta_2$$

Concluding, this procedure is not simple when the focus is on more complex manipulator.

Differential kinematic

Differential kinematics gives the relationship between the joint velocities and the corresponding end-effector linear and angular velocity. This mapping is described by a matrix, termed geometric Jacobian, which depends on the manipulator configuration.
Alternatively, if the end-effector pose is expressed with reference to a minimal representation in the operational space, it is possible to compute the Jacobian matrix via differentiation of the direct kinematics function with respect to the joint variables. The resulting Jacobian, termed analytical Jacobian, in general differs from the geometric one. The Jacobian is one of the most important tools for manipulation; in fact, it is useful for finding singularities, analyzing redundancy, determining inverse kinematics algorithms, describing the mapping between forces applied to the end-effector and resulting torques at the joints and deriving dynamic equations of motion and designing operational space control schemes.

Consider now an $n - DOFs$ manipulator, the direct kinematics equation can be written in the form:

$$T_e(q) = \begin{bmatrix} R_e(q) & p_e(q) \\ \\ 0^T & 1 \end{bmatrix}$$

where $q = [q_1, ..., q_n]^T$ is the vector of joint variables. As said, the goal of the differential kinematics is to find the relationship between the joint velocities and the end-effector linear and angular velocities. So more formally, what is desiderable to do is to express the end-effector lienar velocity $\dot{p_e}$ and angular velocity ω_e as a function of the joint velocities $\dot{q}$.

Compactly we can rewrite, what just said as:

$$\begin{aligned} \dot{p_e} &= J_P(q)\dot{q} \\ \omega_e &= J_O(q)\dot{q} \end{aligned}$$

where, J_P and J_O are the $(3 \times n)$ matrices ralating the contribution of the joint velocities $\dot{q}$ to the end-effector linear velocity $\dot{p_e}$ and end-effector angluar velocity ω_e respectively. So in compact form:

$$v_e = \begin{bmatrix} \dot{p_e} \\ \omega_e \end{bmatrix} = J(q)\dot{q}$$

The above equation represents the manipulator differential kinematics equation, in which, as it is clearly intuible, the$(6 \times n)$ matrix J is the manipulator geometric Jacobian:

$$J = \begin{bmatrix} J_P \\ J_O \end{bmatrix}$$

Taking as reference a generic open kinematic chain and consider a generic link i, as in the following figure:

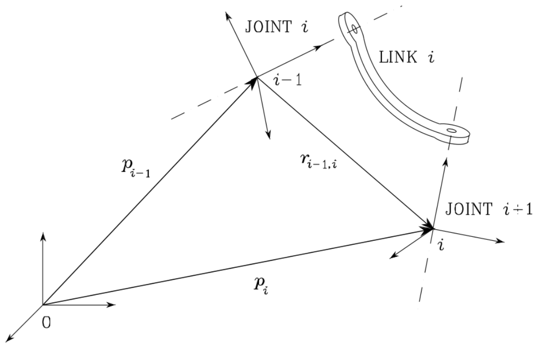


Figure 2.16: Linear velocity of i with respet to $i-1$

According to the Denavit–Hartenberg convention link i connects Joints i and $i+1$, frame i is attached to link i and has origin along joint $i+1$ axis, while frame $i-1$ has origin along joint i axis. Letting p_{i-1} and p_i be the position vectors of the origins of frames $i-1$ and i, respectively. Also, let r_{i-1}^{i-1} denote the position of the origin of frame i with respect to frame $i-1$ expressed in frame $i-1$. According to the coordinate transformation, one can write then:

$$p_i = p_{i-1} + R_{i-1} r_{i-1}^{i-1}$$

which can be rewritten as:

$$\dot{p}_i = \dot{p}_{i-1} + R_{i-1}\dot{r}_{i-1}^{i-1} + \omega_{i-1} \times R_{i-1} r_{i-1,i}^{i-1} = \dot{p}_{i-1} + v_{i-1,i} + \omega_{i-1} \times r_{i-1,i}$$

which gives the expression of the linear velocity of link i as a function of the translational and rotational velocities of link $i-1$.

For what regard angular velocity, it is necessary to start from rotation composition:

$$R_i = R_{i-1} R_i^{i-1}$$

in which, writing the time derivative, we have:

$$S(\omega_i)R_i = S(\omega_{i-1})R_i + R_{i-1}S(\omega_{i-i,i}^{i-1})R_i^{i-1}$$

where ω_{i-1}^{i-1} denotes the angular velocity of frame i with respect to frame $i-1$ expressed in frame $i-1$.

Without diving into the the dimonstration, the angular velocity results:

$$\omega_i = \omega_{i-1} + R_{i-1}\omega_{i-i,i}^{i-1} = \omega_{i-1} + \omega_{i-1,i}$$

which gives the expression of the link i as a function of the angular velocities of link $i-1$ and of link i with respect to link $i-1$.

Consider now, the two main types of joint, for the first one, the prismatic joint, we have that the orientation of frame i with respect to frame $i-1$ does not vary by moving Joint i,, in other words:

$$\omega_{i-1,i} = 0$$

while, the linear velocity is:

$$v_{i-1,i} = \dot{d}_i z_{i-1}$$

where z_{i-1} is the unit vector of joint i axis. On the other hand, for the revolute joint, we have:

$$\omega_{i-1,i} = \dot{\theta}_i z_{i-1}$$

while, for the linear velocity it is:

$$v_{i-1,i} = \omega_{i-1,i} \times r_{i-1,i}$$

due to the rotation of frame i with respect to frame $i-1$ induced by the motion of joint i.

Finally, in order to compute the Jacobian, it is better to start from the linear velocity, the time derivative of $p_e(q)$ can be written as:

$$\dot{p}_e = \sum_{i=1}^{n} \frac{\partial}{\partial p_e} q_i \dot{q}_i = \sum_{i=1}^{n} J_{Pi} \dot{q}_i$$

Each term represents the contribution of the velocity of single joint i to the end-effector linear velocity when all the other joints are still. Distinguishing, the prismatic joint to the revolute one, we have:

- If the joint is prismatic,

$$\dot{q}_i J_{pi} = \dot{d}_i z_{i-1}, \text{ where } J_{Pi} = z_{i-1}$$

- If the joint is revolute,

$$\dot{q}_i J_{pi} = \omega_{i-1} \times r_{i-1,e} = \dot{\theta}_i z_{i-1} \times (p_e - p_{i-1}), \text{ where } J_{Pi} = z_{i-1} \times (p_e - p_{i-1})$$

On the other hand, for what regards the angular velocity, we have:

$$\omega_e = \omega_n = \sum_{i=1}^{n} \omega_{i-1,i} = \sum_{i=1}^{n} J_{Oi} \dot{q}_i$$

Also in this case, it is necessary to make a distinction between prismatic and revolute joint:

- If joint i is prismatic:

$$\dot{q}_i J_{Oi} = 0, \text{ then } J_{Oi} = 0$$

- If the joint is revolute:

$$\dot{q}_i J_{Oi} = \dot{\theta}_i z_{i-1}, \text{ then } J_{Oi} = z_{i-1}$$

Finally, the Jacobian can be written as:

$$\begin{bmatrix} J_{p1} & \cdots & J_{Pn} \\ \vdots & \ddots & \vdots \\ J_{O1} & \cdots & J_{On} \end{bmatrix}$$

where

$$\begin{bmatrix} J_{pi} \\ J_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} for\, a\, prismatic\, joint \\ \\ \begin{bmatrix} z_{i-1} \times (p_e - p_{i-1}) \\ z_{i-1} \end{bmatrix} for\, a\, revolute\, joint \end{cases}$$

The concept of kinematic redundancy has already been introduced, it relates to the number n of $DOFs$ of the structure under analysis, the number m of operational space variables and the minimal number r of operational space variables necessary to perform a task. In order to performa systematic analysis if redundancy it is worth considering differential kinematics in lieu of direct kinematics. To this purpose, is to be interpreted as the differential kinematics mapping relating the n components of the joint velocity vector to the $r \leq m$ components of the velocity vector v_e of concern for the specific task.

So, writing the kinematic equation that has to be considered:

$$v_e = J(q)\dot{q}$$

where now v_e is meant to be the $(r \times 1)$ vector of end-effector velocity, J is the corrisponging $(r \times n)$ Jacobian matrix that can be extracted from the geometric Jacobian and finally $\dot{q}$ is the $(n \times 1)$ vector of joint velocities. As said, if $r < n$, the manipulator is kinematically redundant and there exit $(n - r)$ redundant $DOFs$.

Taking as reference the following set theory figure:

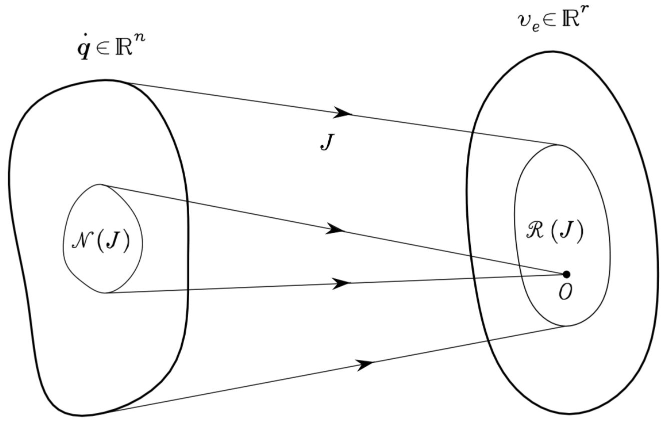


Figure 2.17: Mapping from joint space to end-effector space

The Jacobian describes the linear mapping from the joint velocity space to the end-effector velocity space, however, the Jacobian has to be regarded as a constant matrix, since the instantaneous velocity mapping is of interest for a given posture.

The differential kinematic equation can be characterized from two main spaces, as the above figure shows:

- The range space of J is the subspace *R(J)* in $\mathbb{R}^r$ of the end-effector velocities that can be generated by the joint velocities, in the given manipulator posture.
- The null space of J is the subspace in *N(J)* in $\mathbb{R}^n$ of joint velocities that do not produce any end-effector velocity.

So it is clearly understandable that, if the Jacobian has full rank, the range of J spans the entire space $\mathbb{R}^r$, on the other hand if J is not full rank, the Jacobian degenerates at a singularity.

Up to now, it has been shown the way to compute the end-effector velocity in terms of the velocity of the end-effector frame. The Jacobian is computed according to a geometric technique in which the contributions of each joint velocity to the components of end-effector linear and angular velocity are determined. If the end effector is specified in terms of minimal number of parameters as, $x_e = \begin{bmatrix} p_e & \phi_e \end{bmatrix}$, it is natural to ask whether it is possible to compute the jacobian via differentiationof the direct kinematics function with respect to the joint variables. Then an analytical techinque is presented in order to compute the Jacobian.
The translational velocity of the end-effector frame can be expressed as the time derivative of vector p_e:

$$\dot{p}_e = \frac{\partial p_e}{\partial q}\dot{q} = J_P(q)\dot{q}$$

For what regards the rotational velocity of the end-effector frame, the minimal representation of orientation in terms of three variables ϕ_e can be considered:

$$\dot{\phi}_e = \frac{\partial \phi_e}{\partial q}\dot{q} = J_\phi(q)\dot{q}$$

Computing the latter Jacobian $(J_\phi(q))$ as $\frac{\partial \phi_e}{\partial q}$ is not straightforward, considering that the funtion ϕ_e is not easy available, but requires computation of the elements of the relative rotation matrix. Even these are the premeses, the differential kinematics equation can be obtained as the time derivative of the direct kinematics equation:

$$\dot{x}_e = \begin{bmatrix} \dot{p}_e \\ \dot{\phi}_e \end{bmatrix} = \begin{bmatrix} J_P(q) \\ J_\phi(q) \end{bmatrix} \dot{q} = J_A(q)\dot{q}$$

where the analytical Jacobian

$$J_A(q) = \frac{\partial k(q)}{\partial q}$$

is different from the geometric one since, the end-effector anguar velocity ω_e with respect to the base frame is not given by $\dot{\phi}_e$.

In a more extensive form, in order to clarify, the analytical Jacobian can be written as:

$$J_A(q) = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \dots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial q_1} & \frac{\partial f_m}{\partial q_2} & \dots & \frac{\partial f_m}{\partial q_n} \end{bmatrix}, \; J_A(q) \in \mathbb{R}^{m \times n}$$

Inverse differential kinematics

Normally, problems arise whenever the end-effector attains a particular position and/or orientation in the operational space, or the structure of the manipulator is complex enough and it results difficult to relate the end-effecotor pose to different sets of joint variables, or else the manipulator is redundant.On the other hand, the differential kinematics equation represents a linear mapping between the joint velocity space and the operational velocity space, although it varies with the current configuration. This fact suggests the possibility to utilize the differential kinematics equation to tackle the inverse kinematics problem. Supposing now to have a motion trajectory assigned to the end-effector in term of v_e and also the initial condition on position and orientation. The goal is to determine a feasible joint trajectory $(q(t), \dot{q}(t))$ that reprodices the desired trajectory. So considering the geometrical Jacobian:

$$\dot{q} = J^{-1}(q)v_e$$

if the initial manipulator condition is known, joint positions can be computed though integration:

$$q(t) = \int_0^t \dot{q}(\varsigma)d\varsigma + q(0)$$

The integral can be solved in discrete time using numerical techinques. The simplest one in based on Euler integration method, in whic, given an integration interval Δt, if the joint positions and velocities at time t_k are known, the joint positions at time $t_{k+1} = t_k + \Delta t$ can be computed as:

$$q(t_{k+1}) = q(t_k) + \dot{q}(t_k)\Delta t.$$

This technique for inverting kinematics is independent of the solvability of the kinematic structure. It is necessary, nonetheless, that the Jacobian would be square and of full rank, otherwise we could go into redundancy or, worse, singularity configurations.

2.2 Collaborative robotics

The chance of sharing the workspace is a crucial aspect about deciding a solution. The adoption of a collaborative robot seems to satisfy both worlds, indeed, in one case, there is the necessity to maintain a certain grade of manual skill, on the other side, robots exhibit precision, power and endurance. This precisely of what this section deals, but first, it is necessary to face some terms and definitions, then some notions of collaborative industrial robot system design and application requirements and at the end some basic concepts of safety.

Terms and definitions

Collaborative operation, state in which a purposely designed robot system and an operator work within a collaborative workspace.
Mechanical power, mechanical rate of doing work, or the amount of energy consumed per unit time.
Collaborative workspace, space within the operating space where the robot system (including the workpiece) and a human can perform tasks concurrently during production operation.
Quasi-static contact, contact between an operator and part of a robot system, where the operator body part can be clamped between a moving part of a robot system and another fixed or moving part of the robot cell.
Transient contact, contact between an operator and part of a robot system, where the operator body part is not clamped and can recoil or retract from the moving part of the robot system.
Protective separation distance, shortest permissible distance between any moving hazardous part of the robot system and any human in the collaborative workspace.
Body model, representation of the human body consisting of individual body segments characterized by biomechanical properties.

2.2.1 Collaborative industrial robot system design

The operational characteristics of collaborative robot systems are significantly different from those of traditional robot system installations and other machines and equipment. In collaborative robot operations, operators can work in close proximity to the robot system while power to the robot's actuators is available, and physical contact between an operator and the robot system can occur within a collaborative workspace. Any collaborative robot system design requires protective measures to ensure the operator's safety at all times during collaborative robot operation, then a risk assessment is necessary to identify the hazards and estimate the risks associated with a collaborative robot application (fig. 2.18).

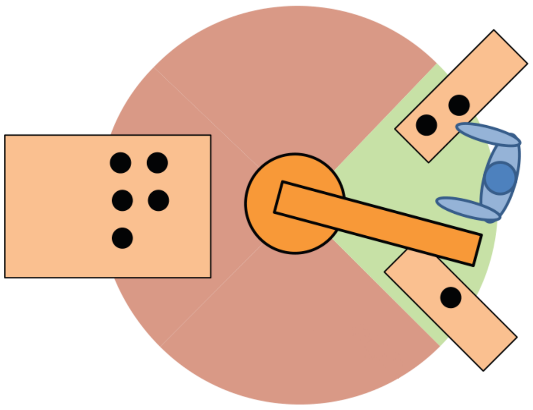


Figure 2.18: Example of collaborative workspace

Design of collaborative application

A key process in the desogn of the collaborative robot system and the associated cell layout is the elimination of hazards and reduction of risks that could influence the design of the working environment. It is for this reason that it is necessary to take into account some considerations:

- *The established limits* of the collaborative workspace;
- *Collaborative workspace*, access and clearance as:
  - delineation of restricted space;
  - influences on the collaborative workspaces (e.g. obtacles, material storage and so on);
  - the need for clearances around obstacles.
  - accessibility for operators;
  - the intended and reasonably foreseeable contact(s) between portions of the robot system and an operator;
  - paths (e.g. taken by the operators, material movement)
  - hazards associated with slips, trips and falls
- *Ergonomics and human inteface* with equipment:
  - clairy and controls;
  - possible stress, fatigue, or lack of concentration arising from the collaborative operation;

- error of misuse by operator;
- possible reflex behaviour of operator to operation of the robot system and related equipment;
- required training level and skills of the operator;
- acceptable biomechanical limits under intended operation and reasonably foreseeable misuse;
- potential conseguences of single or repetitive contacts;

• *Use limits*

  - description of the tasks including the required training and skills of an operator;
  - identification of the tasks including the require training and skills of an operator;
  - potential intended and unintended contact situations;
  - restriction of access to authorized operators only;

• *Transions*, time limit:

  - starting and ending of collaborativve operations;
  - transitions form collaborative operations to other types of operation.

Hazard identification and risk assessment

The list of significative hazards for robot and robot systems shall be addressed on an individual basis through risk assessment for the specific collaborative robot application. The hazard identification processh shall then consider the following as a minimum:

• *Robot related hazards*:

  - robot characteristics;
  - quasi-static contact conditions in the robot;
  - operator location with respect to proximity of the robot;

• *Hazard related to the robot system*:

  - end-effector and workpiece hazards;
  - operator motion and location with respect to positioning of parts, orientation of structures;
  - fixture design, clamp placement and operation

  - determination on whether the contacts would be transient or quasi-static and if the parts of the operator's body could be affected;
  - the design and location of any manually controlled robot guiding device;
  - the influence and effects of the surroundings;

- *Application related hazards*:
  - process-specific hazards;
  - limitations caused by the required use of personal protective equipment;
  - deficiency in ergonomic design.

After that hazards are identified, the risks associated with the collaborative robot system shall be assessed before applying risk reduction measures, so listing these fundamental principles in order of priority:

- the elimination of hazard by inherently safe design or their reduction by substitution;
- protective measures that prevent personnel from accessing a hazard or control the hazards by bringing them to a safe state before an operator can access or be exposed to the hazards.
- the provision of supplementary protective measures such as information for use, training, signs, personal protective equipments and so on.

For traditional robotic systems, risk reductionis typically achieved through safeguards that separate the operator from the robot system. For a collaborative application, the risk reduction is primarly addressed by the design and application of the robot system and of the collaborative workspace.

2.2.2 Requirements for collaborative robot system applications

Due to the potential reduction of the spatial separation of human and robot in the collaborative workspace, phisical human-robot contact can occur during the operation. Protective measures shall be provided to ensure the operator's safety at all times. It is for thie reason that the following requirements shall all be fulfilled:

- *The risk assessment* shall consider the entire collaborative task and workspace, including:
  - robot characteristics;
  - end-effector hazards, including the workpiece (sharp edges, ergonomic design and so on);

  - layout of the robot system;
  - operator location with respect to proximity of the robot arm;
  - operator location and path with respect to proximity of the robot arm;
  - fixture design;
  - design and location of any manually controlled robot guiding device;
  - application-specific hazards;
  - limitations caused by the use of any necessary personal protective equipment;
  - environment considerations;
  - performance criteria of the associated safety functions.

- *Robots integrated* into a collaborative workspace shall meet the requirements of ISO 10218-1.
- Protective devices used for presence detection shall meet the requirements of subsection 5.2 of ISO 10218-2.
- *Additional protective deviced* used in collaborative workspace shall meet the requirements of section 5.2 of ISO 10218-2.
- *The safeguarding* shall be designed to prevent of detect any person from advancing firther into the safeguarded space beyond the collaborative workspace. Intrusion into the safeguarded space beyond the collaborative worlspace shall cause the robot to stop and all hazards to cease.
- *The perimeter* safeguarding shall prevent od detect any person from entering the non-collaborative portion of safeguarded space.
- If *other machines*, which are connected to attached to the robot system and present a potential hazard, are in the collaborative workspace itself then the safety-related functions of these machines shall comply, at a minumin, with the requirements of subsection 5.2 of ISO 10218-2.

Requirements for collaborative workspaces

The design of the collaborative workspace shall be such that the operator can easily perfrom all tasks and the location of equipment and machinery shall not introduce additional hazards. Safe-rated soft axes and space limiting should, whenever possible, be used to reduce the range of possible free motions.
The robot system should be installed to provide a minimum clearance of 500 mm from the operating space of the robot areas of building, structures, utilities, other machines,

and equipment that allow whole body access and may create a trapping oa a pinch point. Where this minimum clearance is not provided, additional protective measures to stop robot motion shall be taken into account, in order to provide protection while personnel are within 500 mm of the trapping or pinch hazard in a static environment.

2.2.3 Collaborative robot system operations

Collaborative operations may include one or more of these methods:

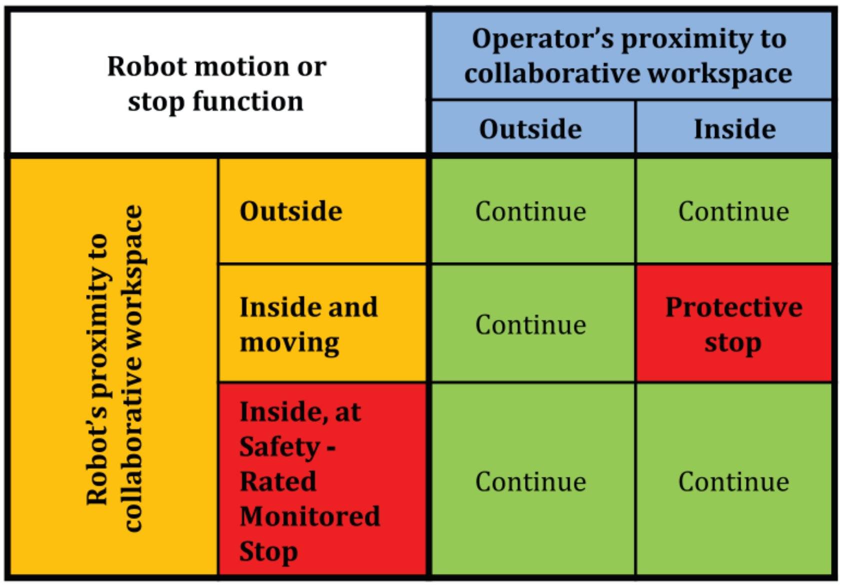
- *Safety-rated* monitored stop;
- *Hand guiding*;
- *Speed and separation* monitorning;
- *Power and force* limiting.

Safety-rated monitored stop

In this method, the safety-rated monitored stop robot feature is used to cease robot motion in the collaborative workspace before an operator enters the collaborative workspace to interact with the robot system and complete a task. If there is no operator in the collaborative workspace, the robot may operate non-collaboratively. When the robot system is in the collaborative workspace, the safety-rated monitored function is active and robot motion is stopped, the operator is permitted to enter the collaborative workspace. Robot system motion can resume without any additional intervention only after the operator has exited the collaborative workspace.
For collaborative operation with safety-rated monitored stop, the following system requirements are needed:

- When robot motion si limited accordingly with:
  - Soft limits are software-defined limits to robot motion while in automatic mode or any mode using speeds above reduced speed;
  - Axis limiting is used to define the restricted space of the robot;
  - Space limiting is used to define any geometric shape which may be used as an exclusionary zone
- The robot shall be equipped with the function to achieve a protective stop

Considering the *safety-rated monitored stop* collaborative method, the robot system The robot system is permitted to enter the collaborative workspace only when an operator is not present in the collaborative workspace. If an operator is not present in the collaborative workspace, the robot system may operate non -collaboratively in the collaborative workspace; following the truth table below, it is possible to have a clearer definition:



Robot motion or stop function		Operator's proximity to collaborative workspace	
		Outside	**Inside**
Robot's proximity to collaborative workspace	**Outside**	Continue	Continue
	Inside and moving	Continue	**Protective stop**
	Inside, at Safety - Rated Monitored Stop	Continue	Continue

Figure 2.19: Truth-table for safety-rated monitored stop method

The robot system shall be equipped with safety-rated devices which detect the presence of an operator within the collaborative workspace. When the sfaety-rated monitored stop feature is used, an operator shall be permitted to enter in the collaborative workspace under the following conditions:

- When the robot system or other hazards are not present in the collaborative workspace;
- When the robot system is in collaborative workspace and it is in safety-rated monitored stop;
- When the robot system is in collaborative workspace in a protective stop condition.

Hand guiding

Under this method of operation, an operator uses a hand-operated device to transmit motion commands to the robot system. before the operator is permitted to enter the collaborative shared workspace and coduct the hand-guiding task, the robot achieves a safety-rated monitored stop. This tipology of task is performed by manually actuating devices loacted at or near the robot end-effector.

Robot systems used with hand-guiding method, could be equipped with additional features, such as, force amplification (force sensors as OPTOFORCE), virtual safety zone and so on.

The requirements for this method imply that the robot shall utilize a safety-rated monitored speed function and safety-rated monitored stop function. A risk assessment shall be used to determine the safety-rated monitored speed limit. If the safety of the operator is dependent on limiting the range of motion of the robot, the robot shall utilize safety-rated soft axis ans space limiting.

The operating sequence for hand guiding is the following:

- The robot system is ready for hand guiding when it enters the collaborative workspace and issues a safety-rated monitored stop;
- When the operator has taken control of the robot system with the hand guiding device, the safety-rated monitored stop is cleared and the operator performs the hand guiding task;
- When the operator releases the guiding device, a safety-rated monitored stop, shall be issued; When the operator ha exited the collaborative workspace, the robot system may resume non-cooperative operation.

If the operator enter in the cooperative workspace before the system is ready for hand guiding, then the protective stop should issue. But it is important to understand that, transitions between hand guiding operations and non-collaborative operation or other types of collaborative operation shall not introduce additional risk.

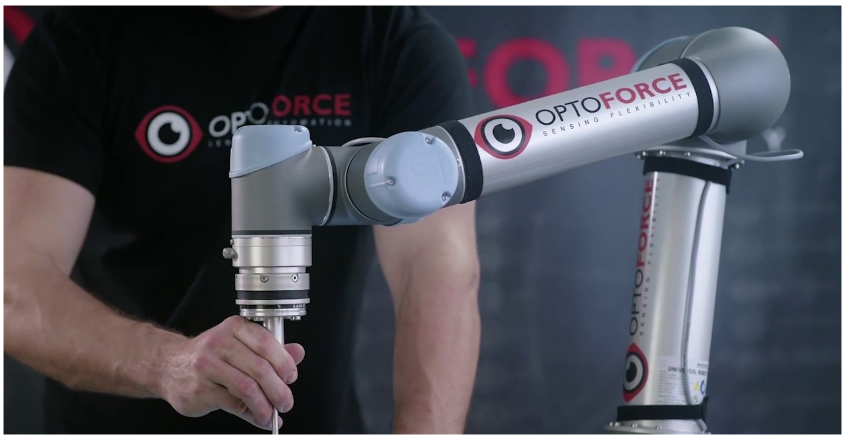


Figure 2.20: Example of hand guiding method

Speed and separation monitoring

The very big difference with respect to the first two is that, under this method, the robot system and operator may move concurrently in the collaborative workspace. Risk

reduction is achieved by maintaining at least the protective separation distance between operator and robot at all times. During robot motion, the robot system never gets closer to the operator than the protective separation distance. When the separation distance decreses to a value below the protective one, the robot systems stops. When the operator moves away from the robot system, the manipulator can resum motion automatically according to the requirements of this clause while maintaining at least the protective separation distance. When the robot system reduces its speed, the protective separation distance decrease corrispondingly.
Also for this method the robot shall be equipped with safety-rated monitored speed function and a safety-rated monitored stop function. If operator safety is dependent on limiting the range of motion of the robot, the robot should be equipped with safety-rated soft axis and space limiting.
Speed and separaion monitoring shall apply to all persons within the collaborative workspace. If the performance of the protective measure is limited by the number of persons in the collaborative workspace then this muaximum numbershall be stated in the information for use. If this value should exceed, a protective stop must occur.
If the separation distance between a hazardous part of the robot system and any operator falls below the protective separation distance, then the robot system shall:

- Initiate a protective stop;
- Initaiate a safety-related functions connected to the robot system.

The possibilities by which the robot control system can avoid violating the protective separation distance include:

- Speed reduction, possible followed by a transition to safety-rated monitored stop;
- Execution of an alternative path which does not violate the protective separation distance, continuing with active speed and separation monitoring.

As said, the maximum persmissible speed and the minimum protective separation distances in an application can be either variable or constant. For the first one, the maximum permissible speeds and the protective separation distances may be adjusted continuously based on the relative spees and distances of the robot systema and the operator. For constant values instead, the maximum permissible speed and the protective separation distance shall be determinated though risk assessment at worst cases over the entire course of the application.

During automatic operation, the hazadrous parts of the robot should never get closer to the operator with respect to the protective separation distance. From ISO 13855, it is possible to calculate the protective separation distance based on the concept used to create the minimum distance formula:

- In constant speed setting situations, the worst-case value for the safety-rated monitored speed of robot is used. The constant limit value should be set as a safety-rated monitored speed in order to ensure the constant limit is not exceeded.
- In variable speed setting situations, the speed of the robot system and of the operator are used to deremine the applicable value fo the protective separation distance at each instant. alternatively, the maximum allowed robot speed can be determinated based on the actual sparation distance between the robot and operator.
- The stopping distance of the robot is determinated according to ISO 10218-1

The protective separation distance S_p, can be computed as:

$$S_p(t_0) = S_h + S_r + S_s + C + Z_d + Z_r$$

Where:

-t_0 is the present or current time;
-S_h is the contribution to the protective separation distance attributable to the operator's change in location;
-S_r is the contribution to the protective separation distance attributable to the robot system's reaction time;
-S_s is the contribution to the protective separation distance due to the robot system's stopping distance;
-C is the intrusion distance, this is the distance that a part of the body can intrude into the sensing field before it is detected;
-Z_d is the position uncertainty of the operator in the collaborative workspace, as measured by the presence sensing device resulting from the sensing system measurement tolerance;
-Z_r is the position uncertainty of the robot system, resulting from the accuracy of the robot position measurement system.

The contribution to the protective separation distance attributable to the operator's change in location can be computed as:

$$S_h = \int_{t_0}^{t_0+T_r+T_s} v_h(t)dt$$

Where T_r is the reaction time of the robot system; T_s is the stopping time of the robot, from the activation of the stop command until the robot has halted, this value is

not constant but rather a function fo the robot configuration; v_h is the directed speed of an operator in the collaborative workspace in direction of the moving part of the robot; t is the integration variable.
If the person's speed is not being monitored, the system design shall assume that v_h is $1,6m/s$ in the direction that reduces the separation distance the most. A constant value for S_h using the estimated human speed can be obtained though:

$$S_h = 1,6 \times (T_r + T_s)$$

The contribution to the protective separation distance attributable to the robot system's reaction time is expresses as:

$$S_r = \int_{t_0}^{t_0+T_r} v_r(t)dt$$

Where v_s is the speed of the robot in the course of stopping, from the activation of the stop command until the robot has halted. In this case, v_s is a function of time and can vary due either the robot's speed or direction changing. The system should be designed taking into account a varying v_s in the manner that reduces the separation distance the most:

- If the robot's speed is not being monitored, the system design should assume that this integral is the ronot's stopping distance in the direction that reduces the separation distance the most;
- If the robot's speed is being monitored, the system deign may use the robot's stopping distance from that speed, applied in the direction that reduce the separation distance the most.

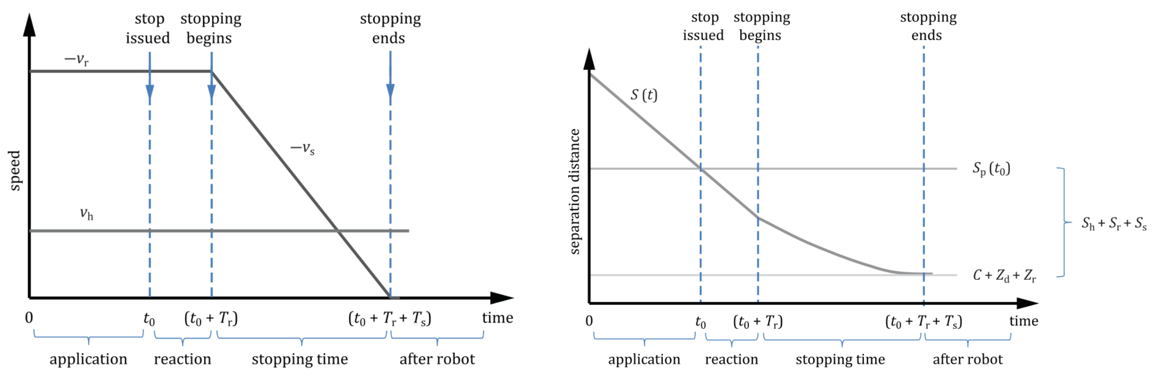


Figure 2.21: Graphical representation of the protective separation distancebetween an operator and robot

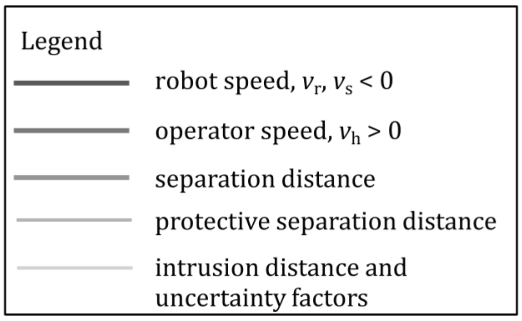

Legend
robot speed, v_r, v_s < 0
operator speed, v_h > 0
separation distance
protective separation distance
intrusion distance and uncertainty factors

Figure 2.22: Legend of the graphical representation of the protective separation distance-between an operator and robot

Power and force limiting

Under this method, physical contact between the robot system (including the workpiece) and an operator can occur either intentionally or unintentionally. Power and force limited collaborative operation requires robot systems specifically designed for this particular type of operation. Risk reduction is achieved, either though inherently safe means in the robot or through a safety-related control system, by keeping hazards associated with the robot system below threshold limit values that are determinated during risk assessment procedure.
During collaborative operations using power and force limiting, contact events between the collaborative robot and body parts of the operator could come about in a number of ways:

- Intended contacts situations that are part of the application sequence;
- Incidental contact situations, which can be a conseguence of not following the correct working procedure, but witjout a techinical failure.
- Failure modes that lead to contacts situations.

Possible contact between moving parts of the robot system and areas on a person's body are categorized as either *quasi-static* contacts, that include clamping or crushing situations or *transient* contacts, that describe a situation in which a person's body part is impacted by a moving part of the robot system and can recoil or retract from the robot without clamping or trapping the contacted body area.

The robot system should be designed to adeguately reduce risks to an operatob by not exceeding the applicable threshold limtit values fow quasi-static and transient contacts, as defined by the risk assessment.
Robot supporting collaborative operations with power and force limiting can be supplied with means to configure limiting thresholds. Risk reduction associated with transient

contacts could involve limiting the speed of moving parts and the appropriate design of the physical characteristics such as the surface area of the moving part that could contact the operator. On the other hand, risk reduction associated with quasi-static contacts could include speed limits and phisical characteristics, similar to transient contacts, plus design characteristics of the parts of the robot system that could involte the possible trappimg or clamping of an operator of the body area. As always, the limit values shall be estimated considering the worst case. These worst case threshold limit values for the transient and quasi-static events shall be used in determing the proper level of risk reduction.

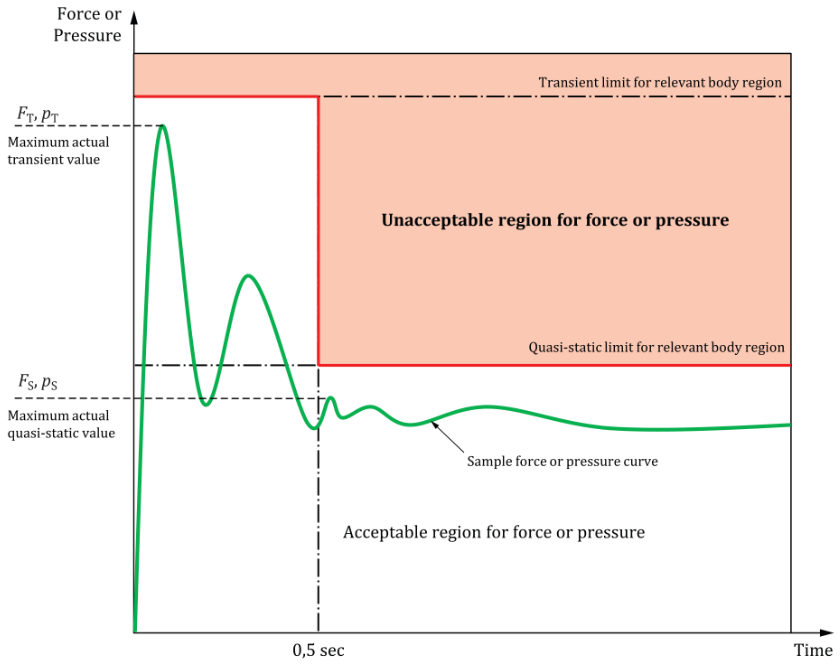

Force or Pressure
F_T, p_T
Maximum actual transient value
F_S, p_S
Maximum actual quasi-static value
Transient limit for relevant body region
Unacceptable region for force or pressure
Quasi-static limit for relevant body region
Sample force or pressure curve
Acceptable region for force or pressure
0,5 sec
Time

Figure 2.23: Graphical representation of force and pressure accettable areas

2.2.4 Relation between biomechanical limits and transfer energy during transient contact

The subject of this part, relates to the computation of the speed in the collaborative workspace. Before considering the model that it will be used for this calculus, it is necessary to specify that under reduced speed control, the speed of the end-effector mounting flange and of the tool centre point shall not exceed 250 mm/sec, in general. More precisely if the collaborative task involves transient contact, the scenario can me modelled taking as reference that, for a contact between a robot and operator, the body contact region and the contact area are known, and the energy transfer can be modified by adjusting the robot velocity at the point of contact. In order to describe this scenario, a simple two-ody model is shown:

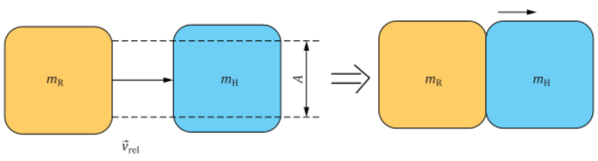


Figure 2.24: Transient contact model

Where:

- A area of contact
- m_H effective mass of human body region
- m_R effective mass of robot as function of robot posture and motion
- v_rel relative speed between robot and human body region

In the model, the robot's effective mass, m_R, shifts to contact the effective mass of the human body region, m_H, at a relative vector velocity, v_{rel}, on a two-dimensional surface, A, resulting in a completely inelastic contact situation, which corresponds, as always, to the worst case hypothesis. The relative kinematic energy is assumed to be fully deposited in the affecected body region. Considering the contact model, m_R can be conservatively estimated as a functiion of the payload capacity of the robot system and the mass of the moving parts of the robot; m_H on the other side, can be estimated as a function of the actual mass of the body region. Also the effective masses and spring constant used to represent the body regions are taken from a table, which are estimated.

For each body region, the maximum permissible energy transferred can be calculates as:

$$E = \frac{F_{max}^2}{2k} = \frac{A^2 p_{max}^2}{2k}$$

Where F_{max} is the maximum contact force, p_{max} is the maximum contact pressure. Once the energy transfer limit value for the contact scenario is established, it can be used to identify the maximum speed at which the robot would be able to move through the collaborative workspace, while maintaining potential pressure and force values below the threshold limits. So, condidering now the general force:

$$E = \frac{F^2}{2k} = \frac{1}{2}\mu v_{rel}^2$$

Where, μ is the reduced mass of the two body system computed as:

$$\mu = \left(\frac{1}{m_H} + \frac{1}{m_R}\right)^{-1}$$

Thus, solving E, it is possible to obtain v_{rel} as and $v_{rel,max}$:

$$v_{rel} = \frac{F}{\sqrt{uk}} = \frac{pA}{\sqrt{uk}} \, and \, v_{rel,max} = \frac{F_{max}}{\sqrt{uk}} = \frac{p_{max}A}{\sqrt{uk}}$$

The contact area is defined by the smaller of the surface areas of the robot of the operator. In situations where the body contact area is smaller than robot contact surface area, such as the operator's hands or fingers, the body contact surface area should be used. Speed limits values, expressed in mm/s, for unconstrained transient contact that can be derived using the body contact model, assuming a contact area of $1cm^2$ are plotted in the following figure.

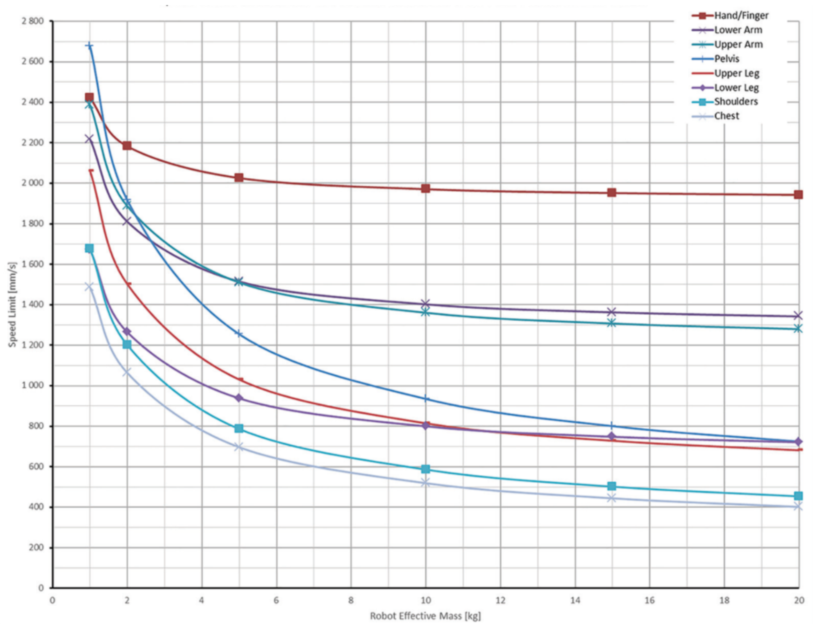


Figure 2.25: Graphical representation of calculated speed limit

2.2.5 Available sensing system technologies

Considering the collaborative robot system operations, it is possible to notice that for the last two methodoligies, there is the necessity to implement sensors that are able to monitor the surrounding environment continuously. These informations will be used to adjust both position and position and dimensions of safety areas. Then 2D and 3D sensors can be used, and each sensor may have different performance in terms of speed, resolution, accuracy. In order to avoid occlusion or to improve safety areas might be covered by several sensors in order to make the environment redundant. The expected vision sensor interaction pattern will be the following:

- Wide-range, low-speed and 2D sensors monitor access to the danger zone at large and medium distance from the robot, to inform downstream sensors of the complexity of the scene;
- Medium-range, faster, 3D sensors monitor the operating space at medium distance from the robot;
- Short-range, high-resolution, very fast, 3D sensors monitor the area at very short distance from the robot;
- Moving objects (men or vehicles) are monitored by tracing them from the peripheral areas of the working area.

As said, there are multiple elements for human detection in the robot workspace and common technologies are listed below, followed by some consideration about pros and cons:

Safety mats

This technology are surface sensors for position detection and typically used to protect hazardous area in industrial environments. Placed all around the workspace to guard they guarantee that nobody is in the dangerous zone, even at the start-up of an application.

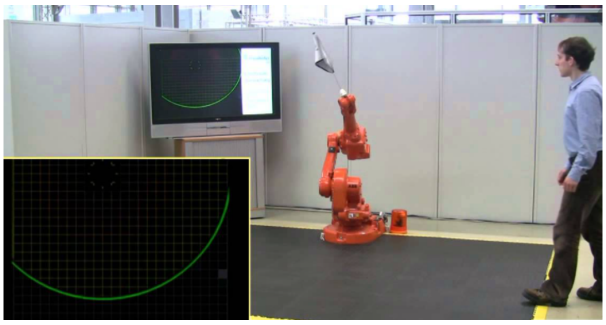


Figure 2.26: Safety mats

-*Pros*: Safety mats are well suited polluted environments since dust and unstable lights conditions do not affects their mechanical measurements.
-*Cons*: They can not be moved once placed so, in case of dynamic application, the room should be covered with mats to ensure safety under all the possible applications, conditions and changes.
-*Output*: Safety mats are divided into cells. When an object is placed on one cell, that cell “actuates”. The return data provided by mats is the indication of the actuated cells.

Stereo Camera

This element is a camera equipped with two (or more) lens that record separate sensor data or video frame, thus simulating the binocular vision of human eyes and therefore capturing 3D images.

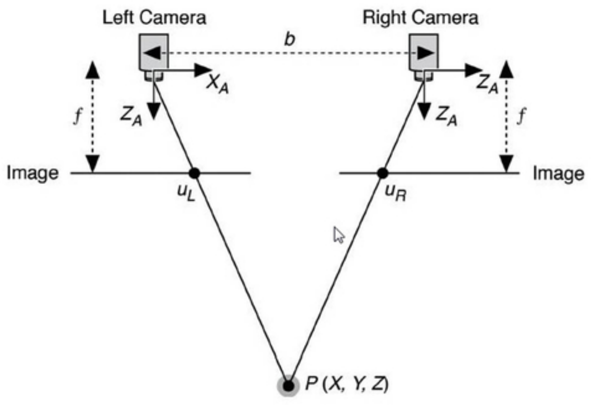


Figure 2.27: Stereo camera scheme

-*Pros*: Stereo cameras are easy to install, have a good speed and spatial resolution and can operate placed in a mid-range distance from the application of interest.
-*Cons*: Stereo cameras are vulnerable to dust, dark or very bright light conditions and to shading caused by the application structure. Clearly, to reduce shading multiple camera systems could be placed at the workspace but with an increased cost due to processing, synchronization and coordination of multiple devices. Furthermore, the collected video images need proper treatment to be compliant with the personal data protection laws.
-*Output*: Via image processing and triangulation. Stereo Cameras provide the distance of every pixel to the next object as data.
Due to the simplicity of installation and to the rapid drop down of the cost of sensors and processors the stereo camera solution is widely used both in industrial and consumer applications.

Laser scanner

Laser scanner is a device able to control the steering of a laser beam to span the surrounding space and rapidly take distance measurements. LiDAR (Light Detection And Ranging) scanners usually offer a two-dimensional area monitoring.

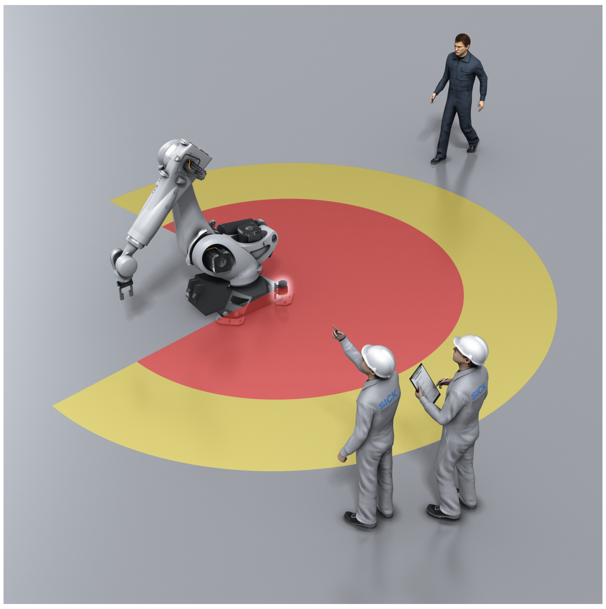


Figure 2.28: Laser scanner

-*Pros*: As stereo cameras, also laser scanners are typically placed in a mid-range distance from the monitored application. However, they are more robust against dust and particles then camera solutions and environmental light conditions do not affect them. In addition, Laser scanners offer a high resolution in the provided angle and range data.
-*Cons*: The motor moving the rotating mirror is more subject to aging and failures than solid state solutions. Shading should always be kept in mind when laser scanners are used for monitoring, as well as the limitation of the horizontal laser beam that does not detect objects above and below the observed plane.
-*Output*: Typically, laser scanners offer a 2D area monitoring (i.e., distances from surface object located through the monitored plane).

Radar system

Radar systems are also devices that compute distances from objects but sending out electromagnetic signals and measure the energy reflected by surrounding environment.

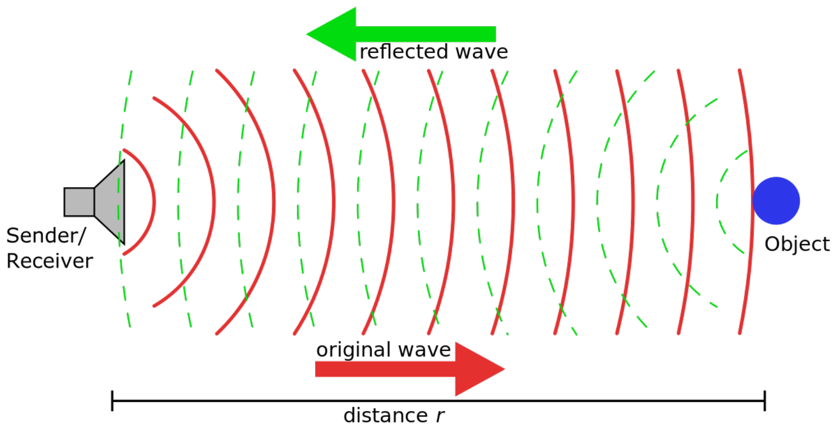


Figure 2.29: Radar system principle

-*Pros*: Radar systems can be used from short to long-range applications (longer ranges result however in a resolution decrease). Radar is very robust against environmental influences but shading e.g., behind metal structures is still possible.
-*Cons*: The function principle of a radar solution leads to an issue when differencing between metal and human obstacles due to their similar energy signature.
Output: Radar provides energy gain as data, which can be transformed to a distance.

Capacitive sensor

Capacitive sensors are detection solutions that measure the distortion of an electric field due to the dielectric properties of an object (e.g., a human body part) that is in the proximity of the sensors.

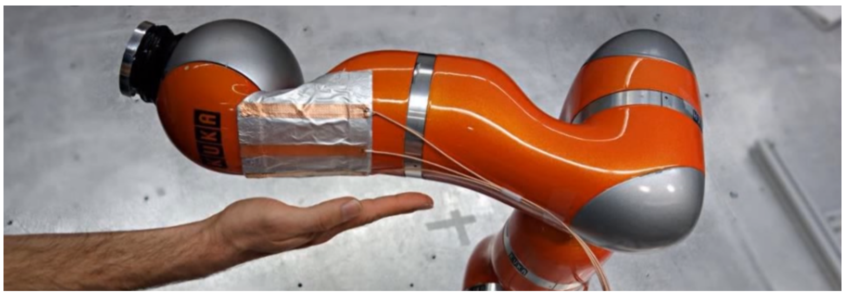


Figure 2.30: Capacitive sensor

-*Pros:* Proximity sensors are well suited for short-range applications, e.g., to serve as a “robot skin”.
-*Cons*: The electric field generated by the capacitive sensors is vulnerable to electromagnetic interferences and humidity changes of the environment.
Output: Capacitive sensors provide the relative change of the electric field as data, which is a value that can be transformed into a distance and/or position of the interfering object.

Pressure sensor

Pressure sensitive sensors are detecting solution that measure an applied force due to the change of resistance other physical principles.

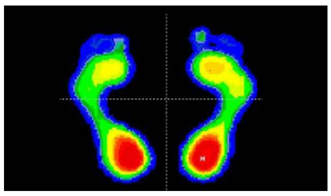


Figure 2.31: Pressure sensor

Pros: This sensors could be used as a tactile skin of the robots for detecting. Pressure sensors are also very robust to environmental conditions.
Cons: A physical contact is always necessary for detection and in many cases (especially those that apply to safety) this characteristic is far to be desirable.
Output: Pressure sensors provide as data both the location of an external object and the pressure that such object applied to the touched sensor.

Ultrasonic sensor

Ultrasonic sensors are solutions based on the measurement of the travel time of an emitted acoustic wave to determine distances from surrounding objects.

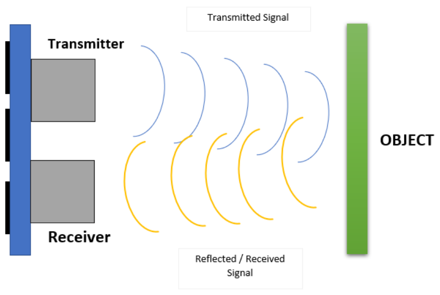


Figure 2.32: Ultrasonic sensor

-*Pros*: Ultrasonic sensors are well suited for detecting transparent surfaces, they are very low cost and not affected by dust or high-moisture environments.
-*Cons*: This sensors are sensible to temperature changes; furthermore, detecting very soft fabrics is difficult because parts of the sound waves can be absorbed.
Output: Provide directly the distance to an object as data.

Time of flight cameras

Time of flight cameras (TOF) are devices that transmit a light impulse to measure the time until the reflected light reaches a sensor mounted on-board the camera itself. Through these measurements flight cameras can reconstruct 3D images of the observed scene (as the stereo cameras seen above).

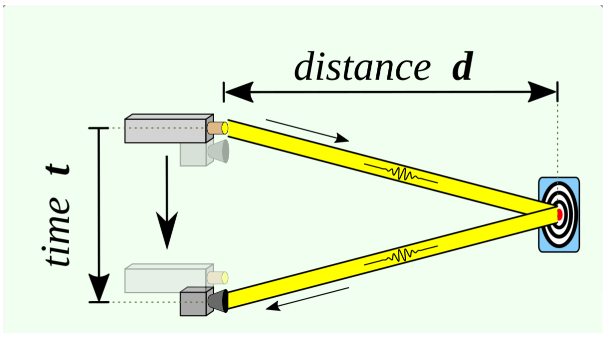


Figure 2.33: TOF camera

-*Pros*: TOF camera does not require the computing power requested by stereo cameras as the sensor outputs directly the distance of the detected surfaces. As they have a single view point the TOF cameras have fewer shading uncertainties on object borders. In principle the could achieve very high frame rates and working distances.
-*Cons*: As all the optical solutions TOF cameras are sensitive to dust and occlusions. The current commercial solutions are sensitive to object reflectance and to high environmental light. If multiple systems coexist in the same workspace, they can interfere with each other ("crosstalk" effect). Multiple light reflections of a single object can also lead to false measurements.
-*Output*: Time of flight cameras capture an image providing distance values for each pixel that compose it.

Chapter 3

Simulation

In this chapter will be illustrated the core of this company thesis, in particular, this chapter introduces the software used to create the environment, and all its details, then it will be shown the robot and it will explained the motivations that led to choose this robot. Then, as said in the introduction, will be listed all the possible scenarios looking for the best one for the cell project. Consequently, will be shown a methodology to represent the closure, trying to minimize the number of parameters and, at the end, will be illustrated the simulation with all the results.

3.1 CoppeliaSim simulation environment

CoppeliaSim is a highly customizable simulator: every aspect of a simulation can be customized. Moreover, the simulator itself can be customized and tailored so as to behave exactly as desired. This is allowed through an elaborate Application Programming Interface (API).
Six different programming or coding approaches are supported, each having particular advantages and also disadvantages over the others, but all six are mutually compatible. In this thesis the approach used in the embedded script:
This method, which consists in writing Lua scripts, is very easy and flexible, with guaranteed compatibility with every other default CoppeliaSim installations. This method allows customizing a particular simulation, a simulation scene, and to a certain extent the simulator itself. The script classification follows the figure 3.1, in which it is possible to distinguish the two major types:

- *Simulation script* are scripts that are executed only during simulation, and that are used to customize a simulation or a simulation model. The main simulation loop is handled via the main script, and models/robots are controlled via child scripts.

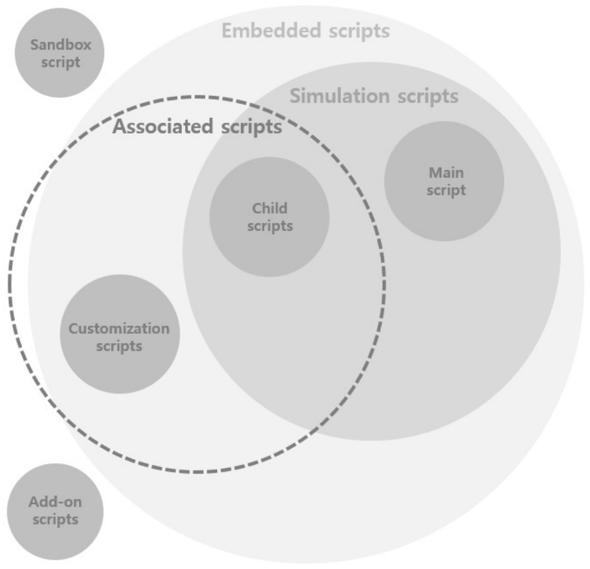


Figure 3.1: Script classification

- *Customization script* are scripts that can also be executed while simulation is not running, and that are used to customize a simulation scene or the simulator itself.

This makes CoppeliaSim very versatile and ideal for multi-robot applications. Controllers can be written in C/C++, Python, Java, Lua, Matlab, Octave or Urbi, but the main scripting language used is Lua, which is an extension programming language designed to support general procedural programming.
CoppeliaSim can be used as a stand-alone application or can easily be embedded into a main client application: its small footprint and elaborate API makes CoppeliaSim an ideal candidate to embed into higher-level applications.
An integrated Lua script *interpreter* makes CoppeliaSim an extremely versatile application, leaving the freedom to the user to combine the low/high-level functionalities to obtain new high-level functionalities.

3.1.1 Dynamic engine

CoppeliaSim's dynamics module currently supports four different physics engines: the *Bullet physics library*, the *Open Dynamics Engine*, *the Vortex Studio engine* and the *Newton Dynamics engine*. At any time, the user is free to quickly switch from one engine to the other according to his/her simulation needs. More precisely:

- *Bullet physics library* is an open source physics engine featuing 3D collision detection, rigid body dynamics, and soft body dynamics. It is largely used in games, and in visual effects in movies.

- *Open Dynamic Engine* is and open source physics engine with two main components: rigid body dynamics and collision detection. It has been used un many applications and games.
- *Vortex Studio engine* is an non-open source, commercial physics engine producing high fidelty physics simulations. Vortex offers real-world parameters for a large number of physical properties, making this engine both realistic and precise. Vortex is mainly used in high performance/precision industrial and research applications.
- *Newton Dynamics* is a cross-platform life-like physics simulation library. It implements a deterministic solver, which is not based on traditional LCP or iterative methods, but possesses the stability and speed of both respectively. This feature makes Newton Dynamics a tool not only for games, but also for any real-time physics simulation.

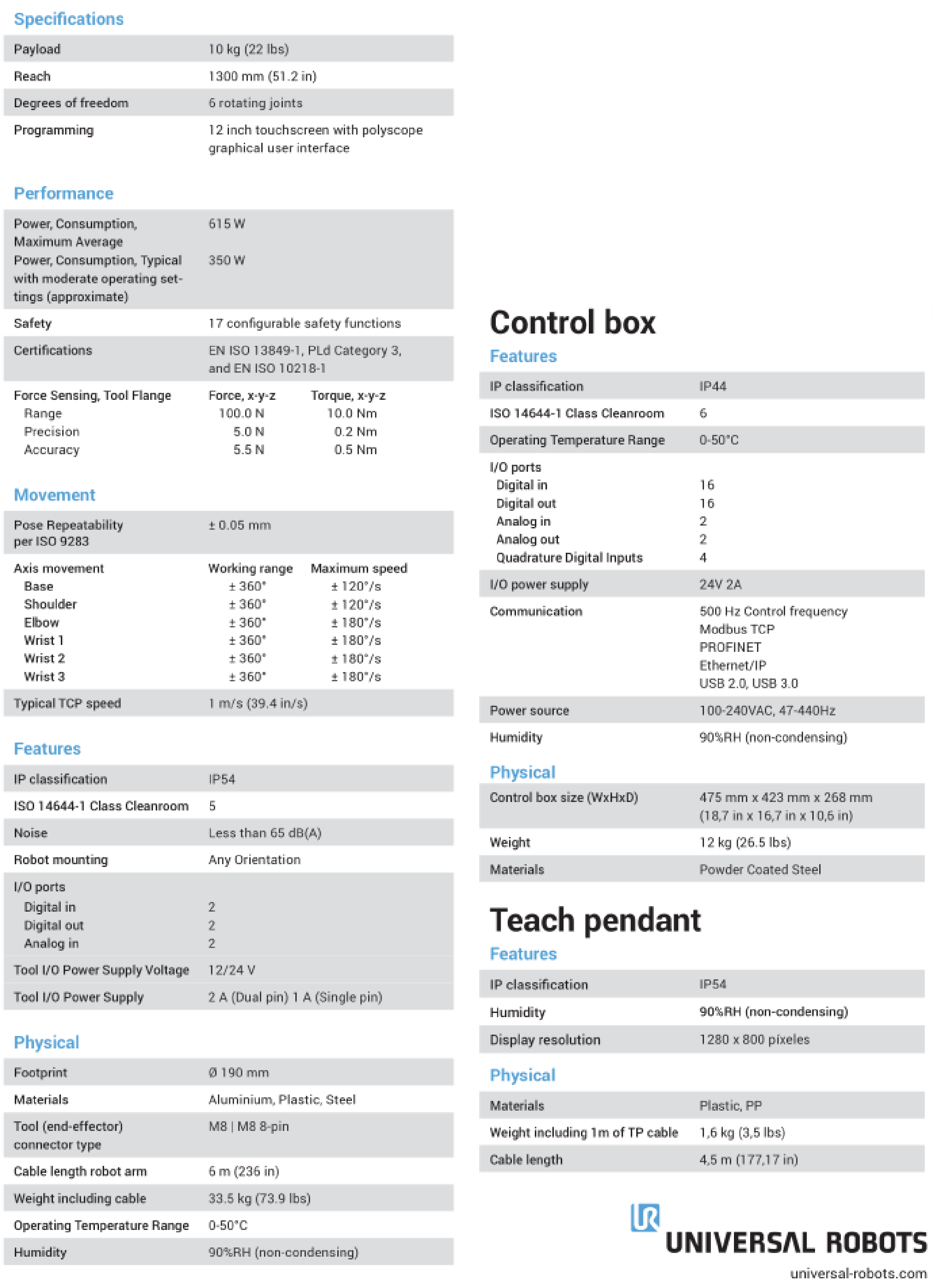
The reason for this diversity in physics engine support is that physics simulation is a complex task, that can be achieved with various degrees of precision, speed, or with support of diverse features. The dynamics module allows simulating interactions between objects that are near to real-world object interactions. It allows objects to fall, collide, bounce back, but it also allows a manipulator to grasp objects, a conveyor belt to drive parts forward, or a vehicle to roll in a realistic fashion over uneven terrain. It is for this reason that, considering the application e the required grade of physics, the *Vortex Studio engine* results to be the best choice.

3.2 Robot choices

Considering the type of application, as mentioned in the abstract, in which the need to have the following characteristics emerges:

- Low frequency;
- High reperability;
- Accuracy;
- Flexibility and reconfigurability;
- The necessity to not create a cell purely dedicated to automation, but to have a shared workspace;

the choice of a collaborative robot turns out to be the optimal one. In particular, from among all the available possibilities that are present, SACMI IMOLA s.c.r.l. focused its attention on a Universal Robot robot, more precisely on the UR10e. On following, it is illustrated the data sheet of the manipulator:



Specifications

Payload	10 kg (22 lbs)
Reach	1300 mm (51.2 in)
Degrees of freedom	6 rotating joints
Programming	12 inch touchscreen with polyscope graphical user interface

Performance

Power, Consumption, Maximum Average	615 W	
Power, Consumption, Typical with moderate operating settings (approximate)	350 W	
Safety	17 configurable safety functions	
Certifications	EN ISO 13849-1, PLd Category 3, and EN ISO 10218-1	
Force Sensing, Tool Flange	Force, x-y-z	Torque, x-y-z
Range	100.0 N	10.0 Nm
Precision	5.0 N	0.2 Nm
Accuracy	5.5 N	0.5 Nm

Movement

Pose Repeatability per ISO 9283	± 0.05 mm	
Axis movement	Working range	Maximum speed
Base	± 360°	± 120°/s
Shoulder	± 360°	± 120°/s
Elbow	± 360°	± 180°/s
Wrist 1	± 360°	± 180°/s
Wrist 2	± 360°	± 180°/s
Wrist 3	± 360°	± 180°/s
Typical TCP speed	1 m/s (39.4 in/s)	

Features

IP classification	IP54
ISO 14644-1 Class Cleanroom	5
Noise	Less than 65 dB(A)
Robot mounting	Any Orientation
I/O ports	
Digital in	2
Digital out	2
Analog in	2
Tool I/O Power Supply Voltage	12/24 V
Tool I/O Power Supply	2 A (Dual pin) 1 A (Single pin)

Physical

Footprint	Ø 190 mm
Materials	Aluminium, Plastic, Steel
Tool (end-effector) connector type	M8 | M8 8-pin
Cable length robot arm	6 m (236 in)
Weight including cable	33.5 kg (73.9 lbs)
Operating Temperature Range	0-50°C
Humidity	90%RH (non-condensing)

Control box

Features

IP classification	IP44
ISO 14644-1 Class Cleanroom	6
Operating Temperature Range	0-50°C
I/O ports	
Digital in	16
Digital out	16
Analog in	2
Analog out	2
Quadrature Digital Inputs	4
I/O power supply	24V 2A
Communication	500 Hz Control frequency Modbus TCP PROFINET Ethernet/IP USB 2.0, USB 3.0
Power source	100-240VAC, 47-440Hz
Humidity	90%RH (non-condensing)

Physical

Control box size (WxHxD)	475 mm x 423 mm x 268 mm (18,7 in x 16,7 in x 10,6 in)
Weight	12 kg (26.5 lbs)
Materials	Powder Coated Steel

Teach pendant

Features

IP classification	IP54
Humidity	90%RH (non-condensing)
Display resolution	1280 x 800 pixeles

Physical

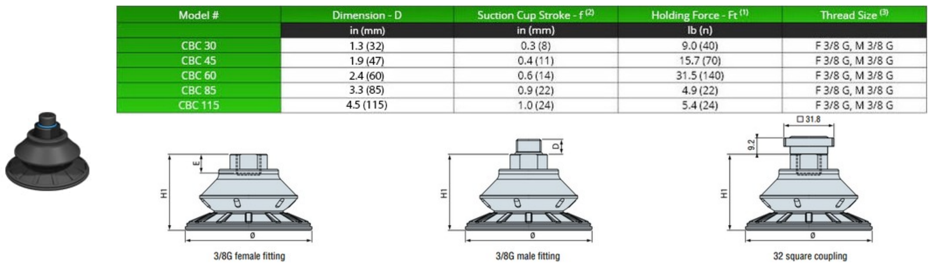
Materials	Plastic, PP
Weight including 1m of TP cable	1,6 kg (3,5 lbs)
Cable length	4,5 m (177,17 in)

UNIVERSAL ROBOTS

universal-robots.com

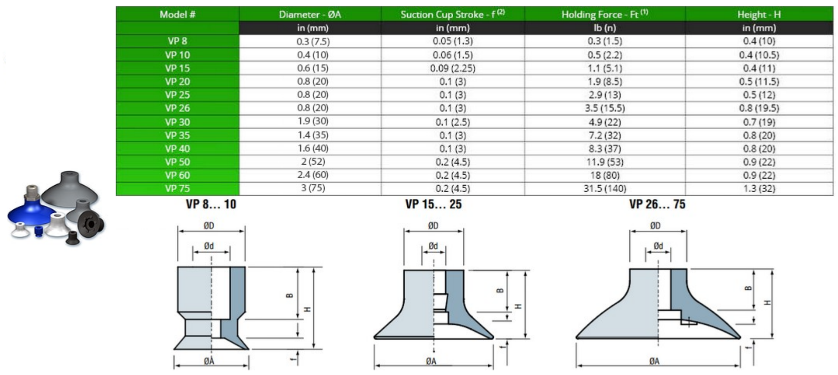
Figure 3.2: Universal Robot UR10e techincal details

The choice of this robot therefore, is motivated by the presence of a very wide range of easily compatible tools. In particular, for this kind of application, in which there is the need to manipulate plastic closures, the company "Millibar Robotics" among the range of products it sells, has a section dedicated to compatible grippers with the collaborative robots of the Universal Robot. More specifically, among these grippers, the choice fell on a suction pad. Here, in following, have been reported some solutions that resulted in line:



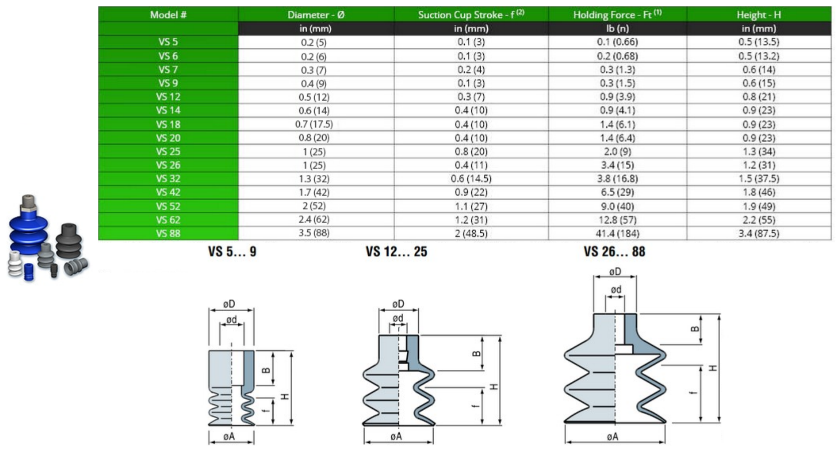
Model #	Dimension - D	Suction Cup Stroke - f (2)	Holding Force - Ft (1)	Thread Size (3)
	in (mm)	in (mm)	lb (n)	
CBC 30	1.3 (32)	0.3 (8)	9.0 (40)	F 3/8 G, M 3/8 G
CBC 45	1.9 (47)	0.4 (11)	15.7 (70)	F 3/8 G, M 3/8 G
CBC 60	2.4 (60)	0.6 (14)	31.5 (140)	F 3/8 G, M 3/8 G
CBC 85	3.3 (85)	0.9 (22)	4.9 (22)	F 3/8 G, M 3/8 G
CBC 115	4.5 (115)	1.0 (24)	5.4 (24)	F 3/8 G, M 3/8 G

E
H1
Ø
3/8G female fitting
D
H1
Ø
3/8G male fitting
□31.8
9.2
H1
Ø
32 square coupling



Model #	Diameter - ØA	Suction Cup Stroke - f (2)	Holding Force - Ft (1)	Height - H
	in (mm)	in (mm)	lb (n)	in (mm)
VP 8	0.3 (7.5)	0.05 (1.3)	0.3 (1.5)	0.4 (10)
VP 10	0.4 (10)	0.06 (1.5)	0.5 (2.2)	0.4 (10.5)
VP 15	0.6 (15)	0.09 (2.25)	1.1 (5.1)	0.4 (11)
VP 20	0.8 (20)	0.1 (3)	1.9 (8.5)	0.5 (11.5)
VP 25	0.8 (20)	0.1 (3)	2.9 (13)	0.5 (12)
VP 26	0.8 (20)	0.1 (3)	3.5 (15.5)	0.8 (19.5)
VP 30	1.9 (30)	0.1 (2.5)	4.9 (22)	0.7 (19)
VP 35	1.4 (35)	0.1 (3)	7.2 (32)	0.8 (20)
VP 40	1.6 (40)	0.1 (3)	8.3 (37)	0.8 (20)
VP 50	2 (52)	0.2 (4.5)	11.9 (53)	0.9 (22)
VP 60	2.4 (60)	0.2 (4.5)	18 (80)	0.9 (22)
VP 75	3 (75)	0.2 (4.5)	31.5 (140)	1.3 (32)

VP 8… 10
VP 15… 25
VP 26… 75
ØD
Ød
B
H
f
ØA



Model #	Diameter - Ø	Suction Cup Stroke - f (2)	Holding Force - Ft (1)	Height - H
	in (mm)	in (mm)	lb (n)	in (mm)
VS 5	0.2 (5)	0.1 (3)	0.1 (0.66)	0.5 (13.5)
VS 6	0.2 (6)	0.1 (3)	0.2 (0.68)	0.5 (13.2)
VS 7	0.3 (7)	0.2 (4)	0.3 (1.3)	0.6 (14)
VS 9	0.4 (9)	0.1 (3)	0.3 (1.5)	0.6 (15)
VS 12	0.5 (12)	0.3 (7)	0.9 (3.9)	0.8 (21)
VS 14	0.6 (14)	0.4 (10)	0.9 (4.1)	0.9 (23)
VS 18	0.7 (17.5)	0.4 (10)	1.4 (6.1)	0.9 (23)
VS 20	0.8 (20)	0.4 (10)	1.4 (6.4)	0.9 (23)
VS 25	1 (25)	0.8 (20)	2.0 (9)	1.3 (34)
VS 26	1 (25)	0.4 (11)	3.4 (15)	1.2 (31)
VS 32	1.3 (32)	0.6 (14.5)	3.8 (16.8)	1.5 (37.5)
VS 42	1.7 (42)	0.9 (22)	6.5 (29)	1.8 (46)
VS 52	2 (52)	1.1 (27)	9.0 (40)	1.9 (49)
VS 62	2.4 (62)	1.2 (31)	12.8 (57)	2.2 (55)
VS 88	3.5 (88)	2 (48.5)	41.4 (184)	3.4 (87.5)

Figure 3.3: Suction pad technology

Successively through a test session, it can be carried out a selection in order to decide which solution turns out to be the correct one.

3.3 Closure's standardization

As said at the beginning of the 1.1 section, the closure production can be made though CCM48 with different dimension or form. At the simulation level it is necessary, in order to generate an identity document, to define a sequence of "key" parameters that allow to identify the model without necessary passing though the 3D representation. This process or better methodology that trivially it is called "Standardization", is the link that connects the ease of description with the compleaxity of the real one enclosing all the variants under a family of variables. Here is what previously explained:

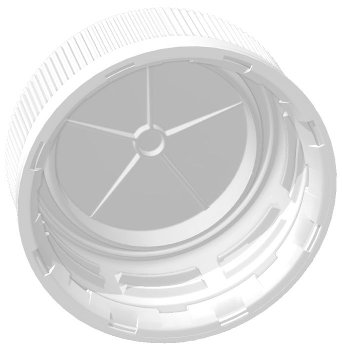


Figure 3.4: 3D closure representation

Exploiting the 3D model it is possible to pass from the above image to the scheme below:

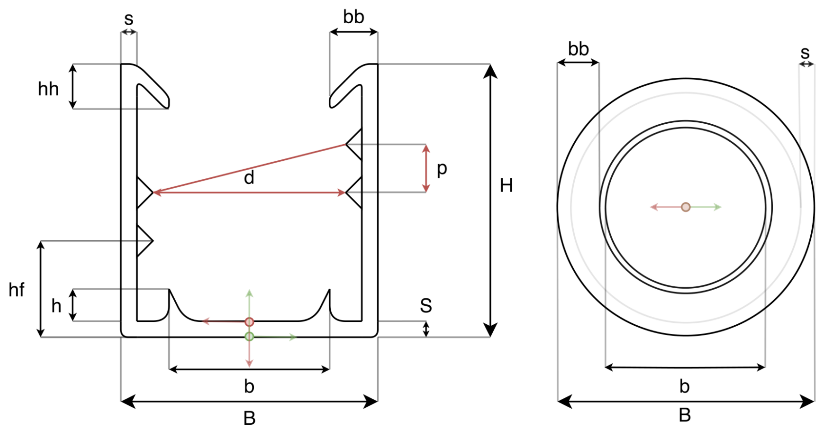


Figure 3.5: Closure standardization

where:

- S is the thickness of the base;
- s is the thickness of the wall;
- B is the outer diameter;
- b is the plug seal diamater;
- bb is the fin lenght;
- H is the height from the base to the top;
- h is the plug seal height;
- hh is the fin height;
- p is the thread step;
- d is the tip-tip thread diameter;
- *angle* is the arctan between p and d;
- *hf* is the height from the base to the first principle.

As it is clearly visible, we passed from a big number of parameters required for the complete definiton to a smaller one.
In the next sections, each parameter will be well defined and it will be clearer why this procedure is crucial in the study.

3.3.1 Introduction to the analysis position

Starting from the the possible production errors explained in the section 1.2, without going into too much details, it is possible to perform a complete analysis of the product defining thirteen analysis positions. What is meant for "analysis position"?
An analysis position is a combination of position and orientation with respect to a well defined reference point.
Today this reference point is the vision camera that the operator uses in order to make the inspection.
These thirteen position can be clustered together in two big group, dividing them according to which lap they belong:

1. Lap
   - Thread

- Plug seal outside
- Plug seal inside
- Leakage burrs

2. Lap

   - Fractures on the fin and unclompleted fin
   - Leakage burrs on the top of the closure and on the fin
   - Top and lateral seal deformation
   - External contamination from oils
   - Not optimal colourant distribution
   - Excessive concavity or convessity of the closure's base

As it is possible to see, for the second lap, we regrouped for the second time considering that it is possible to take the same photo to inspect both of the analysis positions.

3.4 Cyclograms and cell design

Macro-cyclogram

In order to have a better overview, it is necessary first, define the macrotasks that compose the process following the scheme below:

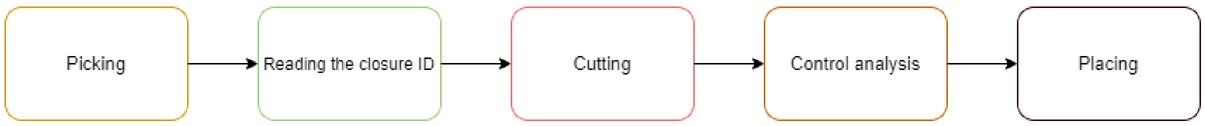


Figure 3.6: Macrotasks

Starting from the left, once the closures have been put in the case, the
-*Picking* phase starts: in this step, the system using a vision sensor, detects the position and orientation of the products. Once these informations has been acquired, through a certain logic, the software decides which closure has to be grabbed.
-*Reading the closure ID* phase: in this part of the process, the system has to understand which piston of the automatic machine has formed the cap under analysis. Therefore, this part of the program is necessary becouse allows, once reached the placing phase, to allocate the closure in the right location of the buffer.
-*Cutting*: as already seen in the problem description chapter, the caps have to be cut in order to complete the analysis procedure. The cutting phase, actually, could not be located only in this position of the program flow, but could be present also after the

control analysis phase.
-*Control analysis*: in this step of the flow chart, there is the core of this process. In this part, the software performs all the inspections that the operator desires, making use of some tools that will be explained later.
-*Placing*: in this last step of the process, there is the sorting of the products. Having read the closure ID from the second step of the cyclogram, it is possible to know which piston made the cap and, in this way, it is the possible to allocate the piece in the right location of the buffer.

3.4.1 Cell design

Having in mind the general process structure, it is possible to start thinking of a probable cell design.

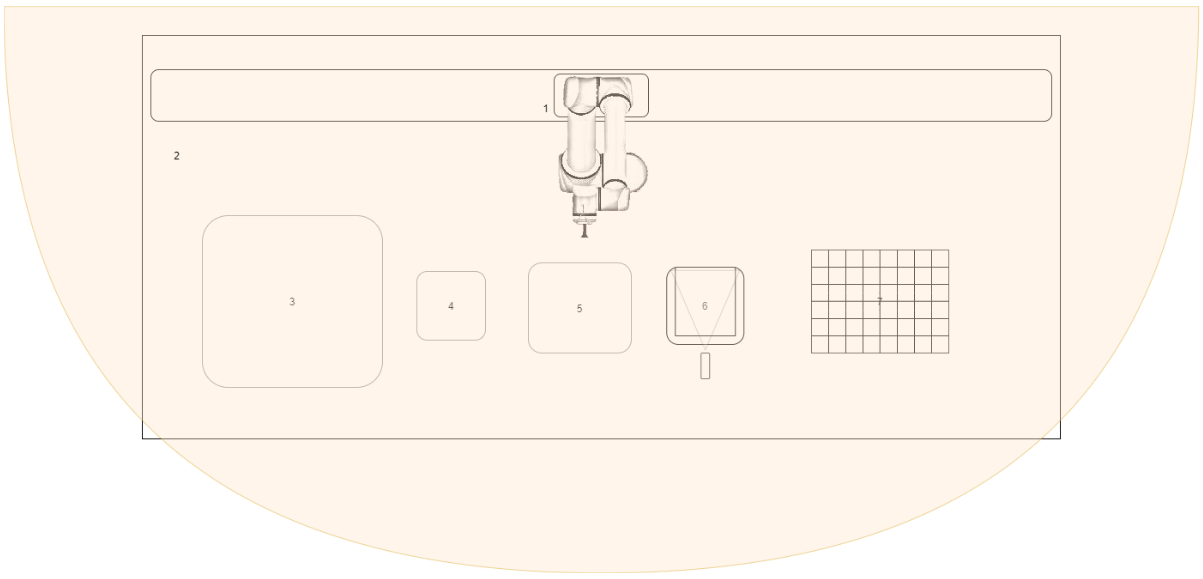


Figure 3.7: Idea of cell design

Taking as reference the above figure, can be distinguished 7 main components:

- 1: Represents the manipulator, in this case it is the representation of the UR10e;
- 2: Represents the ideal workspace;
- 3: Represents the case in which the operators put the closures;
- 4: Represents the bin. This component is necessary in those cases in which the ID read can not be performed due to a bad molding;

- 5: Represents an ideal cutting station;
- 6: Represents an ideal analysis platform;
- 7: Represents the buffer in which the closure are allocated in a sorted way.

Each component defined above, is mandatory for the success of this project. It is for this reason that, in the following subsection, the task will be analyzed in a deeper way, trying not only, to define the main components, but also analyzing in a more detailed way each aspects of the process.

Chosen mode

In order to make an efficient cell design, the following parameters have been taken into account:

- number of vision sensor;
- encumbrance of components;
- vision sensor mounting location;
- operation time;
- operational logic.

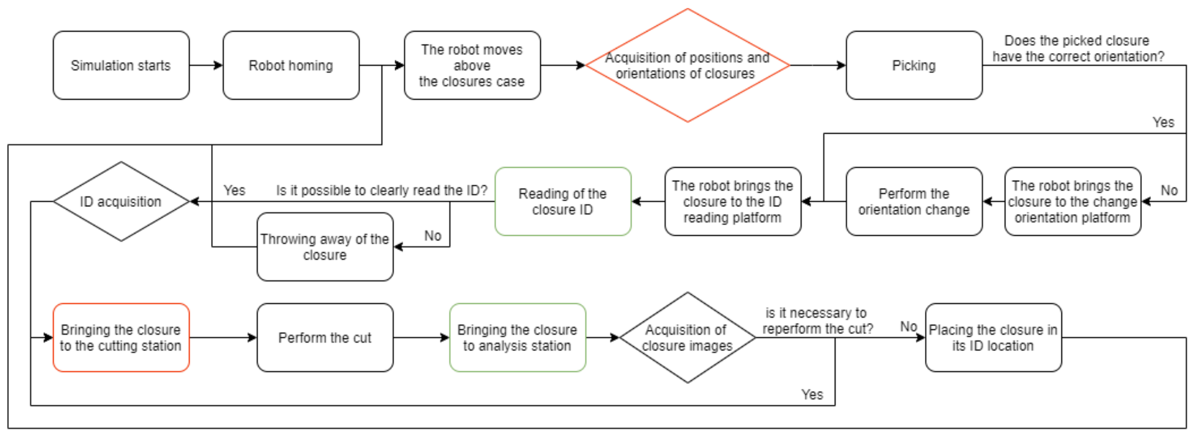

Simulation starts
Robot homing
The robot moves above the closures case
Acquisition of positions and orientations of closures
Picking
Does the picked closure have the correct orientation?
Yes
No
The robot brings the closure to the change orientation platform
Perform the orientation change
The robot brings the closure to the ID reading platform
Reading of the closure ID
Is it possible to clearly read the ID?
Yes
No
Throwing away of the closure
ID acquisition
Bringing the closure to the cutting station
Perform the cut
Bringing the closure to analysis station
Acquisition of closure images
is it necessary to reperform the cut?
No
Yes
Placing the closure in its ID location

Figure 3.8: Flowchart of the mode

In the search for the solution, it has been considered the *Pareto's concept*. What Pareto states is: *in the search for an optimal solution, in which the resources amount is* *predefined, the optimal solution is the solution that does not make any Pareto improvements to the system, in other words, that can not improve the condition of a parameter without damaging another parameter.* Hence, considering the decision parameters as an unique system, Seven possible modes were identified, in which one with respect the others was the best. The figure 3.8 shows the flowchart of the optimal mode, in which the total number of vision sensors are three and in particular, the vision sensor that is in charge to take the position and orientation of the closures in the fourth step of the flowchart, is mounted integrally on the last link of the manipulator. Hence, it is through the following figure (3.9) that the definitive cell design is illustrated:

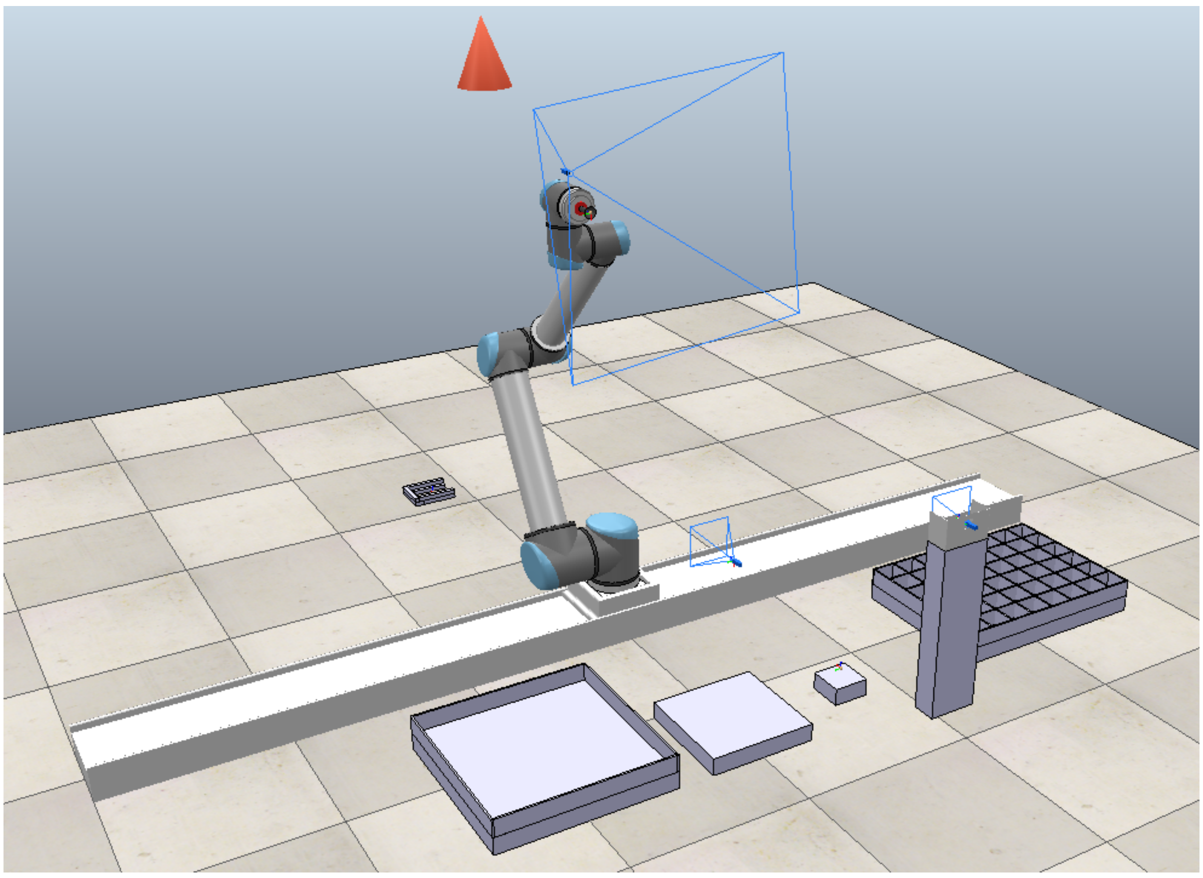


Figure 3.9: Cell design

3.5 Inverse kinematic and frames definition in CoppeliaSim

In CoppeliaSim in order to perform the inverse kinematic, it is necessary first define 2 main entities:

- A reference frame attached to the robot;
- A target reference frame;

these two entities in particular, once the application is running, they will be for all the simulation long always connected and overlapped.
Attaching then the first reference frame to the robot in the desired point, it is possible to decide at which level of the kinematic chain perform the inverse kinematic process. Considering that it is not possible to shift the first reference frame becouse attached in a precise point along the chain, in order to perform the motion of the manipulator, it is necessary then play with the target reference point. In this way, defining for instance a path, and letting the target reference point walks on this, it is possible to perform with the end-effector the desired trajectory.

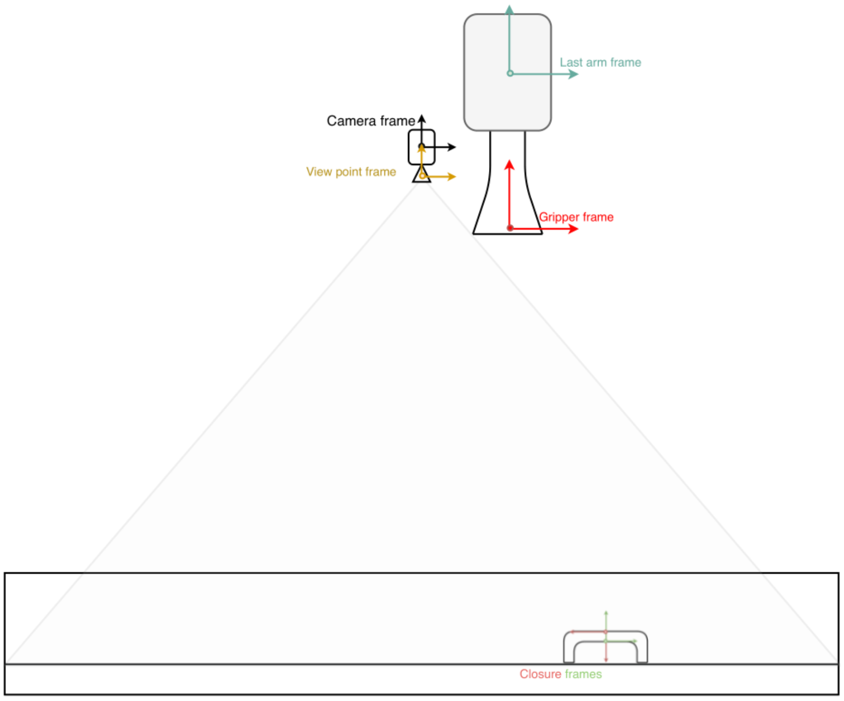


Figure 3.10: Homing kinematic chain

At this point, in order to declare all the frames that have been used during the simulation, it is useful defining all the kinematic chain that have been made, through

the help of the flowchart defined in figure 3.8. Starting then from the homing, in which the robot brings himself above the case, the vision sensor set to the manipulator, acquires the position and orientation of the closures. In order to do it, it is necessary to define a goal reference frame with respect to the case of coordinates $(0, 0, 500)[mm]$.
Making use of the homogeneous transformation matricies and defining the first reference frame and the target reference frame attached to the vision sensor of the robot, defining a path that goes from the position of these two frames to the goal reference frame, it is possible to perform the homing. The frame definition is schematized with the figure 3.10. So more formally, this kinematic chain is composed as:

$$T_W^{VS} = T_W^{Base} \times T_{Base}^{link1} \times T_{Link1}^{Link2} \times \cdots \times T_{LinkN-1}^{LinkN} \times T_{LinkN}^{VS}$$

The position and orientation that the vision sensor acquires, actually, are defined with respect to the vision sensor. In order to make those informations suitable also for the robot, in the picking process, it is necessary to compute the homogeneous transformation matrix of the closure with respect to the world istead of with respect to the vision sensor. So formally, as for the previous case, the homogeneous transformation matrix of the closure with respect to the world is:

$$T_W^{Closure} = T_W^{Base} \times T_{base}^{link1} \times T_{Link1}^{Link2} \times \cdots \times T_{LinkN-1}^{LinkN} \times T_{LinkN}^{VS} \times T_{VS}^{Closure}$$

Going on, following the flowchart, the picking phase comes. In this situation, it is necessary to move the two main frames, defined before, to the gripping frame. In order to do it, it is sufficient, instead of recompute the whole kinemaatic chain, to move only the last part of the computation. In this way, it becomes:

$$T_W^{Gripper} = T_W^{Base} \times T_{base}^{link1} \times T_{Link1}^{Link2} \times \cdots \times T_{LinkN-1}^{LinkN} \times T_{LinkN}^{Gripper}$$

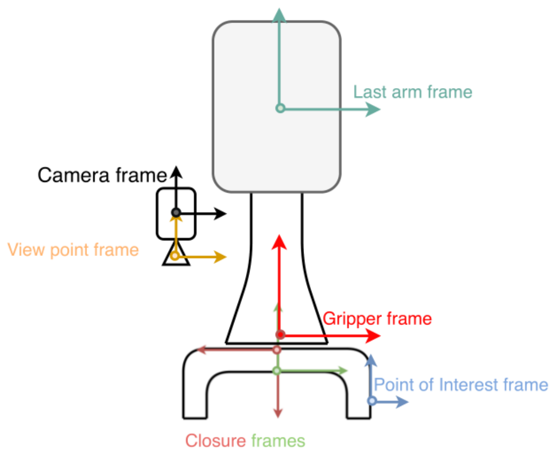


Figure 3.11: Picking phase

in which $T_W^{Closure}$ in this case, is used as goal reference frame.
After the picking phase, it is not unsual to have a non perfect acquisition of the exact position and orientation of the closures, considering that these informations comes from a real component that normally could be affected from noise or errors. So it is easy that the following configuration happens:

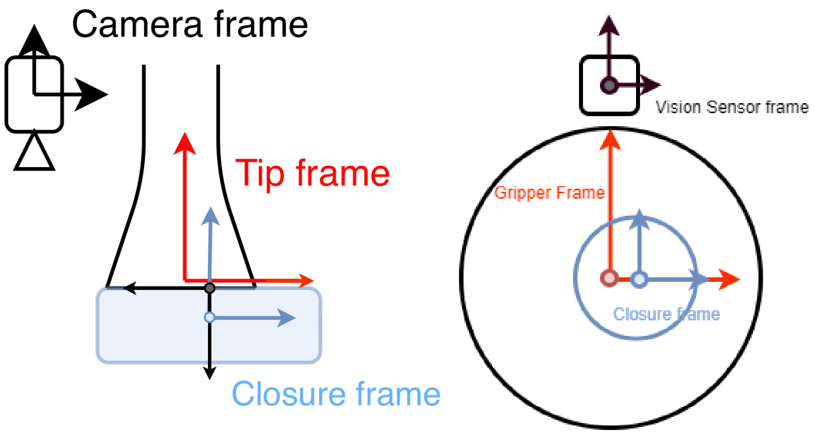


Figure 3.12: Bug in the picking phase

In this situation, there is the presence of a misalignment between the closure frame and the gripper frame that can cause a difficulty in the ID reading phase and therefore a false bad closure. In order to correct this unpleasant behaviour, what can be done is to define another kinematic chain. This kinematic chain exploits the second vision sensor used for the ID reading in order to compute $T_W^{Closure}$ as:

$$T_W^{Closure} = T_W^{VS_2} \times T_{VS_2}^{Closure}$$

This homogeneous transformation matrix will be passed to the manipulator in order to move its two reference frames, used to perform the inverse kinematic, correcting the previous created problem.

Considering the figure 3.8, where it has been defined the flowchart of the process, it is possible to notice that the fourth and the twelveth step have a red contour, while the ninth and the forteenth steps have a green countour. The use of these two colours highlights, in the red case, those steps in which it could be possible to have an error of misalignment, on the other case for the green countours, those steps in which it is possible to perform a correction action as previously explained. It is therefore important to notice that in the subsequent steps of the red one, where there could be a bug creation, the flowchart logic, allows always corrective action in all those steps where the accuracy is a crucial factor.

3.6 Simulation managing

In the first part of the section will be introduced the method that it has been used to control the simulation. In particular first will be explained the scheme in which the simulation is organized then will be shown the Graphic User Interface and the modalities that this latter allows to implement. Starting from the organization, the simulation is structured as:

- Main program flow;
- Graphic User Inteface;
- Closure associated child scripts.

Each simulation component is characterized by a proper associated child script, that could be either threaded or non-threaded.

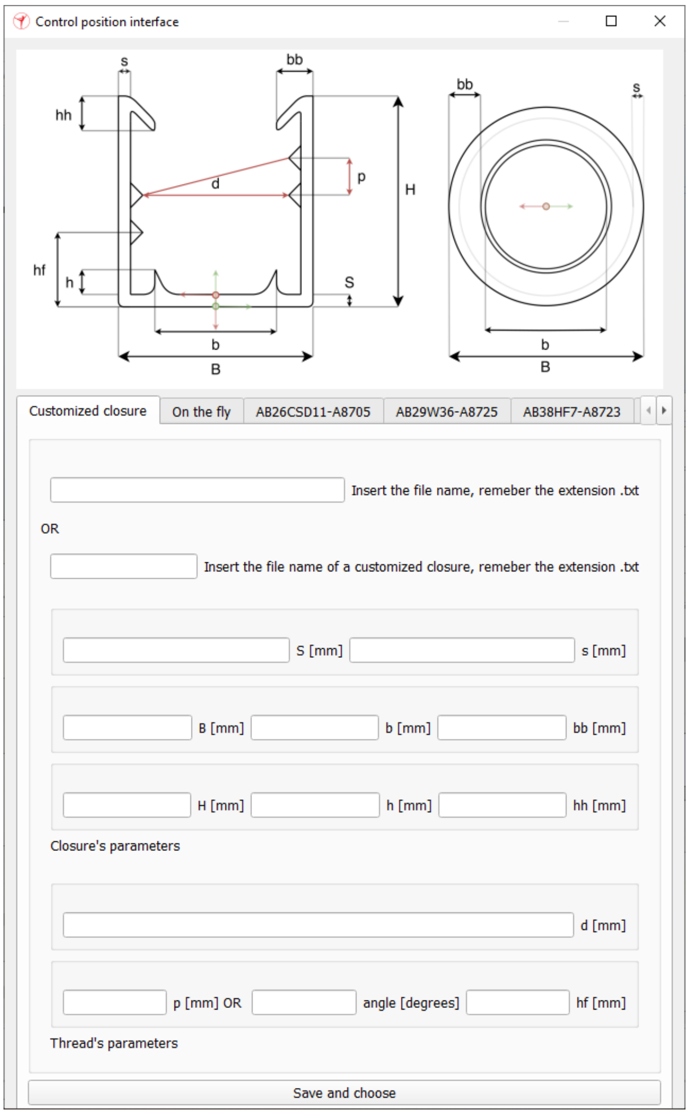
Non-threaded child scripts: Non-threaded child scripts should contain a collection of system callback functions. Those should not be blocking. This means that every time they are called, they should perform some task and then return control. If control is not returned, then the whole simulation halts. Non-threaded child script functions are called at least twice per simulation step from the main script's actuation and sensing functions. Non-threaded child scripts should always be chosen over threaded child scripts whenever possible.

Threaded child scripts: Threaded child scripts are scripts that will launch in a thread. Launch/resume of threaded child scripts is executed in a precise order. When a threaded child script's execution is still underway, it will not be launched a second time. In particular *Threaded child scripts* have a behaviour that is similar to an interpretated language, for this reason, threaded child scripts have several weaknesses compared to non-threaded child scripts if not programmed appropriately: they are more resource-intensive, they can waste some processing time, and they can be a little bit less responsive to a simulation stop command.

In particular the kind of script chosen for each component is:
The main program flow script has been written into a *threaded child-script*, using developed *utilityLibrary.lua* library, because have been used some functions that work only in this kind of script.
The graphic user interface, on the other hand, has been written into a *non-threaded child script* in XML language.
For what regard the *closure associated child scripts* have been written using *non-threaded child scripts.*
Each simulation part will be explained in a more detailed way in the following sections starting from the Graphic User interface, touching the closures generation logic and at the end will be discussed the core of this work, the quality control process.

3.6.1 Graphic User Interface

The libraries written in order to implement the GUI can be downloaded under the name of *GraphicUserIntefrace.lua* and *UserInterface.lua*, following the link at the bottom of the abstract.



Insert the file name, remeber the extension .txt

OR

Insert the file name of a customized closure, remeber the extension .txt

S [mm] s [mm]

B [mm] b [mm] bb [mm]

H [mm] h [mm] hh [mm]

Closure's parameters

d [mm]

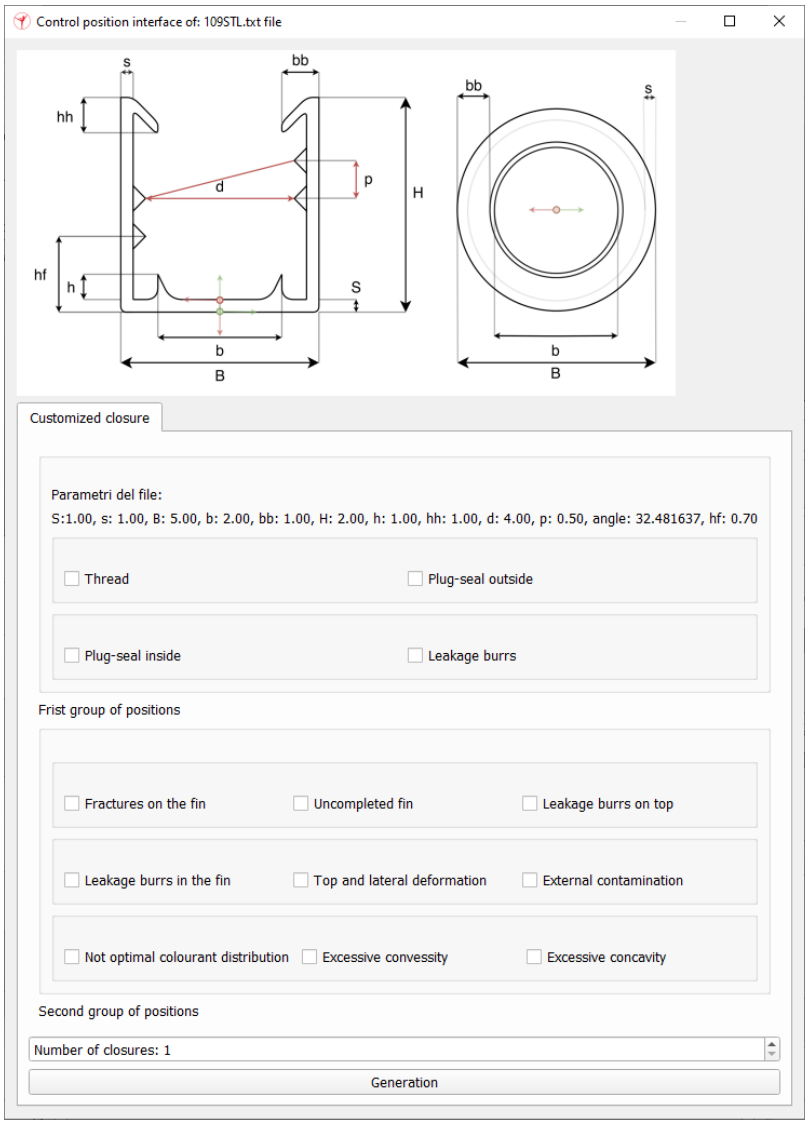
p [mm] OR angle [degrees] hf [mm]

Thread's parameters

Figure 3.13: GUI

Through figure 3.13 it is shown the GUI, that is how the simulation is managed. The GUI is composed of 3 main subgroups, called tab, structured in the following way:

First tab - Customized closure



Parametri del file:
S:1.00, s: 1.00, B: 5.00, b: 2.00, bb: 1.00, H: 2.00, h: 1.00, hh: 1.00, d: 4.00, p: 0.50, angle: 32.481637, hf: 0.70

☐ Thread ☐ Plug-seal outside

☐ Plug-seal inside ☐ Leakage burrs

Frist group of positions

☐ Fractures on the fin ☐ Uncompleted fin ☐ Leakage burrs on top

☐ Leakage burrs in the fin ☐ Top and lateral deformation ☐ External contamination

☐ Not optimal colourant distribution ☐ Excessive convessity ☐ Excessive concavity

Second group of positions

Number of closures: 1

Generation

Figure 3.14: Quality control positions

In the first tab, the operator has the possibility of:

- Create a new customized closure and save it in the data-base;
- Call an already created customized closure, present in the data-base.

If the operator desires to create a new closure, it is sufficient to respect the following procedure:

-*Insert the file name, with the .txt extension and insert the closure parameters following the image shown.*

On the other hand, if the operator desires to recall an already created customized closure, it is just necessary to :
-*Insert the name of the file, followed by the extension .txt.*

Once chosen which operation perform, the GUI will open an other window letting the operator decide which quality-control position perform. The second window it is shown with the figure 3.14. Once decided the number of closure that it is desiderable to generate, through the bottom part of the GUI, it is possible to generate the proper customized closure and let the simulation runs. Obviously at the simulation level it is not possible to design a CAD of the closure, but it is still possible to create a representation of the wanted piece. The following image represent an example of the *109STL.txt*:

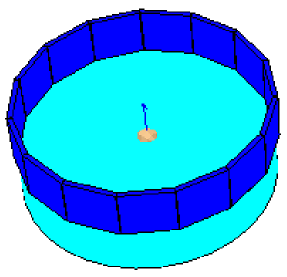
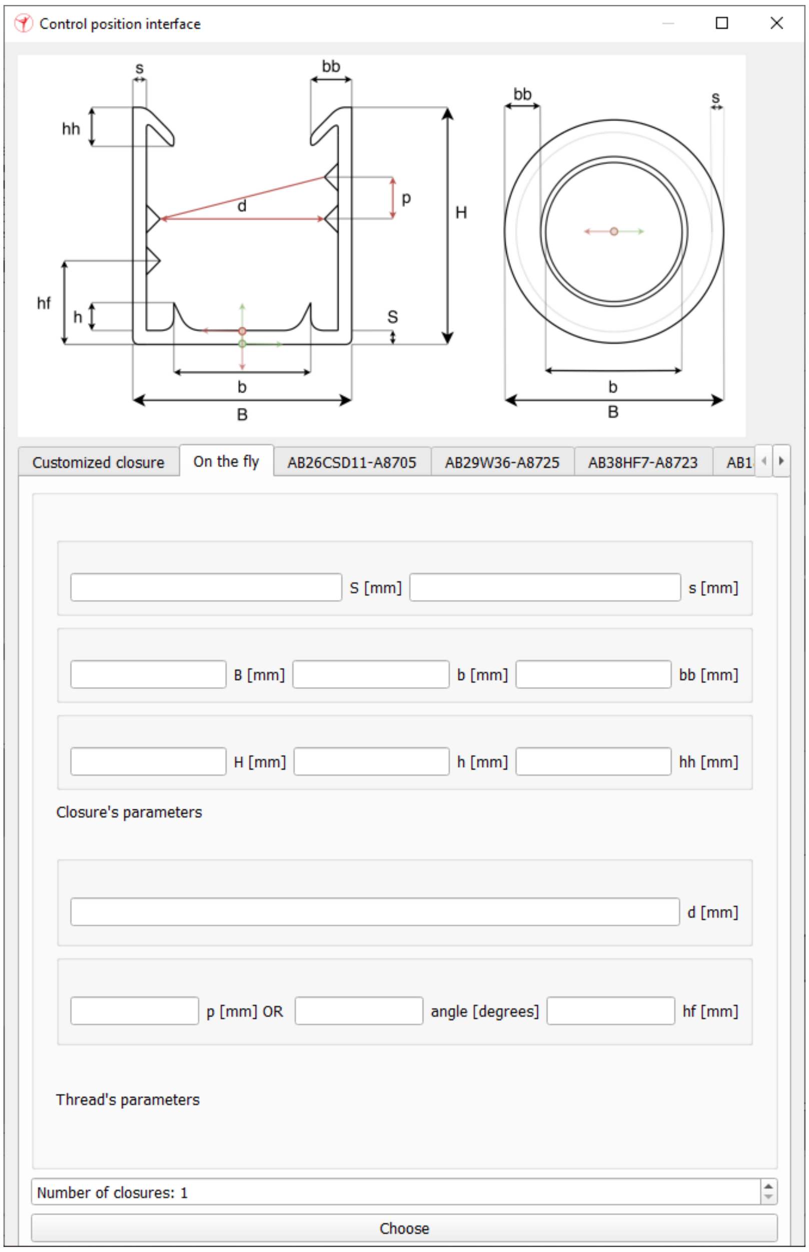


Figure 3.15: Customized closure

Second tab



S [mm] s [mm]

B [mm] b [mm] bb [mm]

H [mm] h [mm] hh [mm]

Closure's parameters

d [mm]

p [mm] OR angle [degrees] hf [mm]

Thread's parameters

Number of closures: 1

Figure 3.16: On the fly closure

The logic behind the managing of this tab is little more complex with respect to the first one. This tab has been implemented for all those situations in which the operator desires to still perform the quality control, but he does not have all the necessary parameters to complete describe the piece. It is easy to understand that, each quality control position requires a certain set of parameters. Therefore, using the fugure 3.17 as legend, it is possible to clarify which parameters are necesessary of each quality control position. At the simulation level however, in order to at least define a closure as rapresented with figure 3.15, S, B and H are the minimum parameters required.

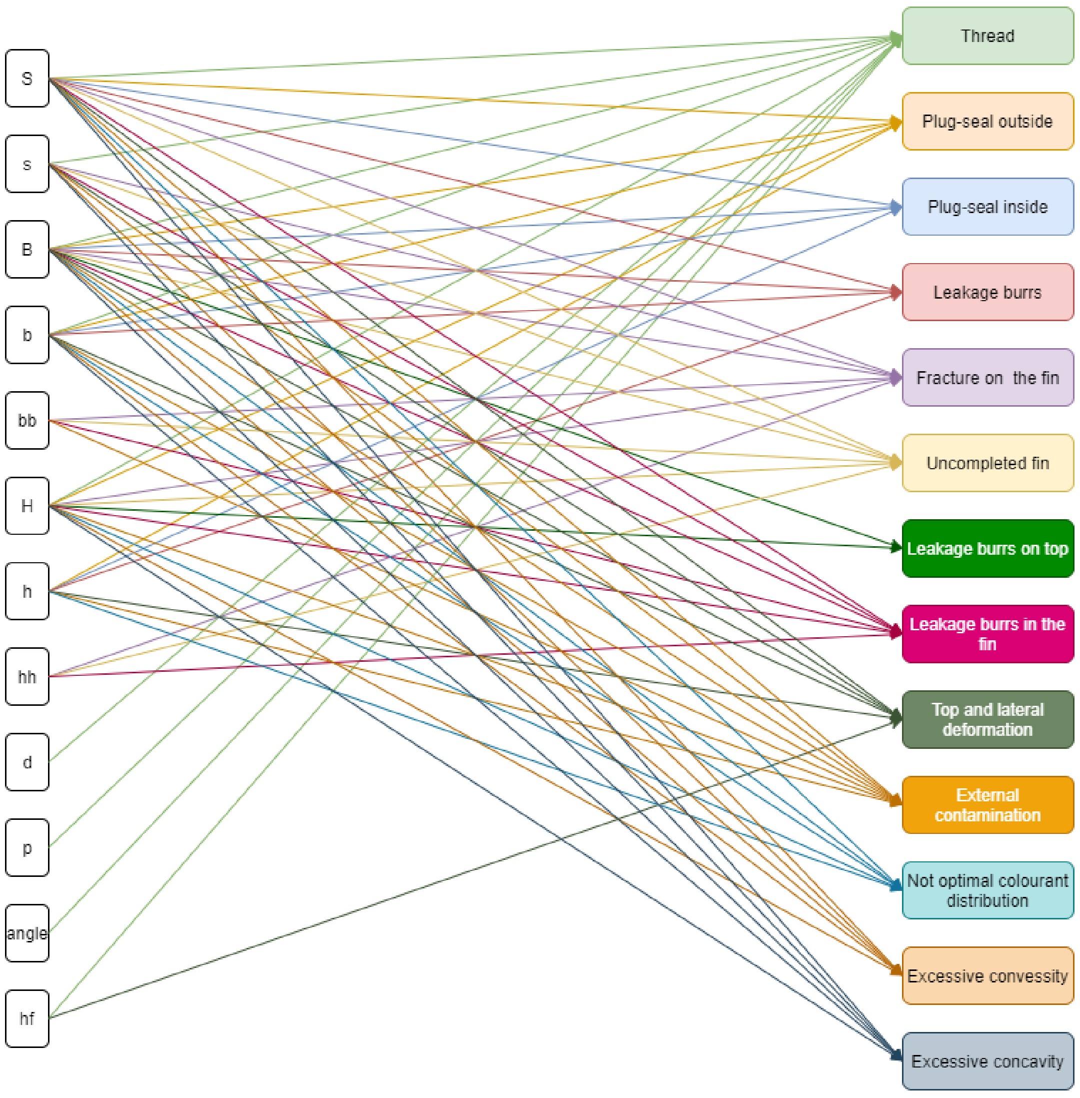
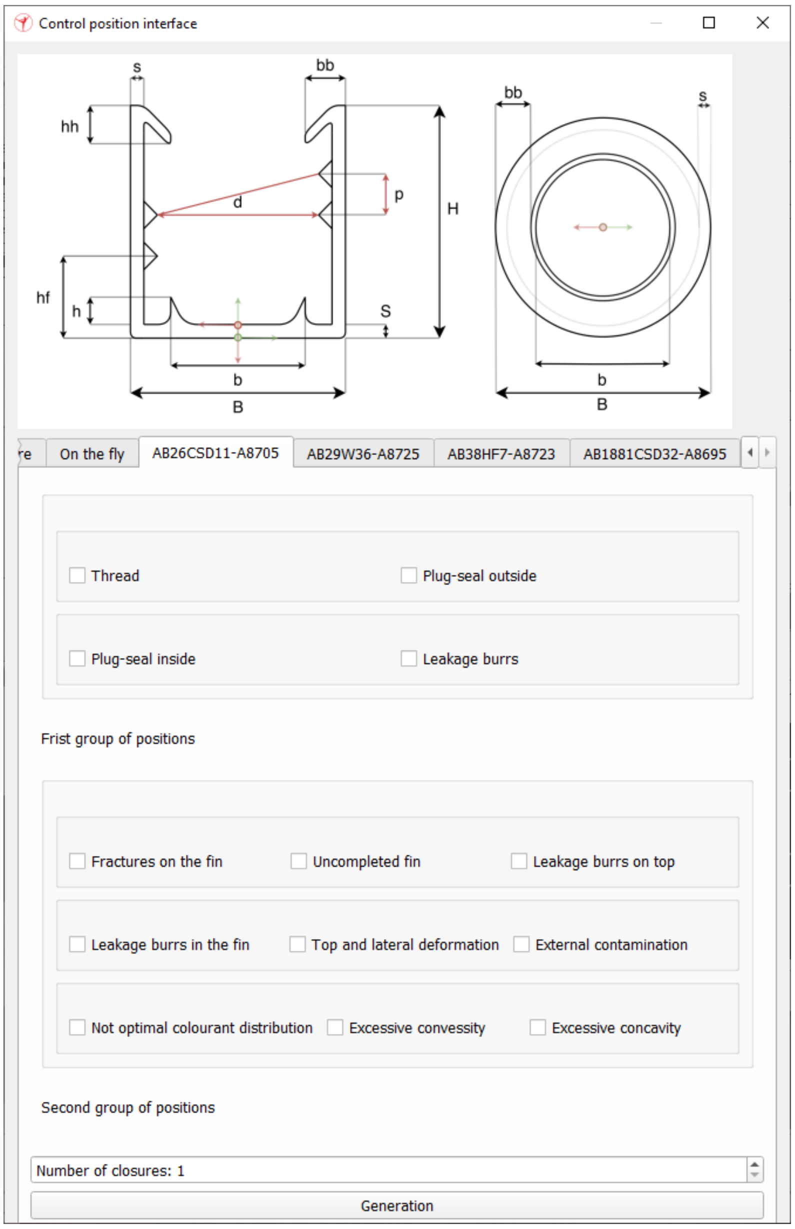


Figure 3.17: Legend for on the fly

Once the parameters are inserted, a second window, as showns in figure 3.14, appears. This time, however, every time that a quality control position is checked, the logic behind the GUI monitors the inserted parameters. If turns out that, for the chosen quality control position, some parameters miss, then a dialog window pops up notifying which parameters the simulation require to perform the desired control.

Others tab

In all these las cases in which, the operator does not want to create a new customized closure, becouse already present in the database and flagged as *standard*, the GUI non only generates the closure in the desired number, but also imports the predefined mesh of the model. In the following, finally, are been shown these las tabs and an example of the imported mesh.


Control position interface

re | On the fly | AB26CSD11-A8705 | AB29W36-A8725 | AB38HF7-A8723 | AB1881CSD32-A8695

- [ ] Thread
- [ ] Plug-seal outside
- [ ] Plug-seal inside
- [ ] Leakage burrs

Frist group of positions

- [ ] Fractures on the fin
- [ ] Uncompleted fin
- [ ] Leakage burrs on top
- [ ] Leakage burrs in the fin
- [ ] Top and lateral deformation
- [ ] External contamination
- [ ] Not optimal colourant distribution
- [ ] Excessive convessity
- [ ] Excessive concavity

Second group of positions

Number of closures: 1

Generation

Figure 3.18: Standard closure tabs

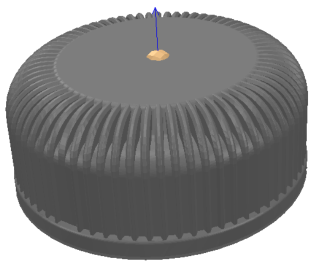


Figure 3.19: Imported mesh

In CoppeliaSim, so at the simulation level, an imported mesh is a very heavy dynamical object computationally. For this reason, instead of considering the dynamic of the mesh, what it has been done was to remove the dynamic properties of this latter, maintaining the visive component and adding an invisible an easier dynamical substructure. So the obtained result is therefore the melt of these two components where, one has been used for computations and the other has been used as visive mask.

3.6.2 ID-reading

In this second part of the simulation managing section, will be explained the way in which the ID-reading phase works and the logic used to avoid the simulative difficulties. Following the flowchart, after the picking phace, comes the ID-reading phase. As said in the 3.4 section, this part deals with the comprehension of which piston created the closure. In the real case, the piston, pressing the plastic into a mould, make the form of the piece, printing also all those codes allowing the operator to understand which piston made this latter. At simulative level, however, even if possible through the available vision sensor, it is not feasible to read the code because this would imply to have all the 48 CAD, one for each piston. In order to avoid this problem, it is been thought of creating the closure and to attach to each closure created an associated non-threaded child script. But since the attached child-script is external to the script that is currently running, the ID-reading alghorithm needs to create a sort of publisher-subscriber channel with which communicate with other scripts. This procedure is implemeted using a string that will be written inside the associated attached child-script. So once the generation button is clicked, independently from which closure has been decided of being created, the pseudo-code logic is:

So once the robot brings the picked closure in front of the vision sensor, and after having performed the correction action explained in the 3.5 section, the vision sensor simulates to read the code inside but actually, receives the informations contained in the non-threaded child script associated.
In particular into each associated non-threaded child script, there is an array that contains two main informations:

- Closure ID;
- Goodness of the closure, used to simulate the quality of the inner code.

These two variables are generated randomically and allow to simulate therefore, random ID code and random quality of the closure. As said, the first variable will be used,

Algorithm 1 Closure generation and ID-reading

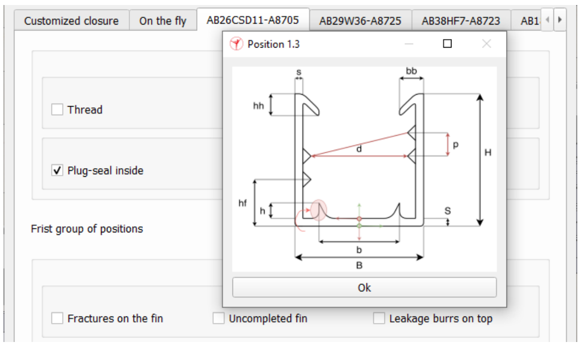
```
1: initilize array
2: initialize string code
3: for $i$ : 1 : $numberOfClosures$ : 1 do
4:     createClosure function
5:     createScript function
6:     Write the string into the script
7:     Associate the script i to the closure i
8: end for
9: readInformation function, Read the data from the channel created inside the associ-
   ated script i when the robot reached the vision sensor
```

following the flowchart, to understand to which location of the buffer place the closure. The second one will be used in order to decide if the closure has to be discarded.

3.6.3 Quality-control

In this subsection, will be faced the quality control problem and the way in which it is managed through the GUI. The developed library for this part can be downloaded, as for the previous two, following the name *positionAnalysis.lua*.

Considering the flowchart of figure 3.8, after the cutting phase, the simulation reaches the quality control phase. It is still through the Graphic User Interface that the operator can decide to which part/s of the closure perform the images acquisition. Taking as reference the second window defined in 3.14 or 3.18 respectively for a customized closure or for a standard closure, the operator can decide che control quality procedure, simply checking the *checkboxes*.



☐ Thread

☑ Plug-seal inside

Frist group of positions

☐ Fractures on the fin ☐ Uncompleted fin ☐ Leakage burrs on top

Figure 3.20: Control position pop-up

Through the previous figure is shown how, everytime that a *checkbox* is checked, a

mini-notifying image pops-up, making the operator sure of the correctness of the procedure. Once the process has been decided, the last task that the GUI is in charge to perform is to control all the positions checked and to create an *array of positions*. Since the GUI, and therefore, the *positionArray* are currently in a script that is running separately with respect to the program main flow, there is, also in this case, the necessity to create a publisher-subscriber channel,making the two scripts communicating. Hence, once the array of positions is passed, the main program flow has all the data that it requires.
Hence, essentially this process is structured in the following way:

```
1: The robot brings the closure in front of the third vision sensor
2: Performing of the corrective action
3: for $i : 1 : numberOfInterestPoints : 1$ do
4:     Compute the interest point frame
5:     Motion of the robot
6: end for
```

In the first step of the loop, the procedure exploits as goal reference frame, a frame called *ControlPlatformFrame* placed at 5 *cm* from the third vision sensor, while the other two frames utilized to perform the inverse kinematic are placed in the clousure interest point that has to be controlled. The reference figure is the 3.11. For the second step, instead, it is computed the path that the two inverse kinematic frames have to walk to bring the interest point frame to the *ControlPlatformFrame*. At this point then the image acquisition can occurs. With the following figure will be shown, as example, the plug-seal control position in the proper control platform.

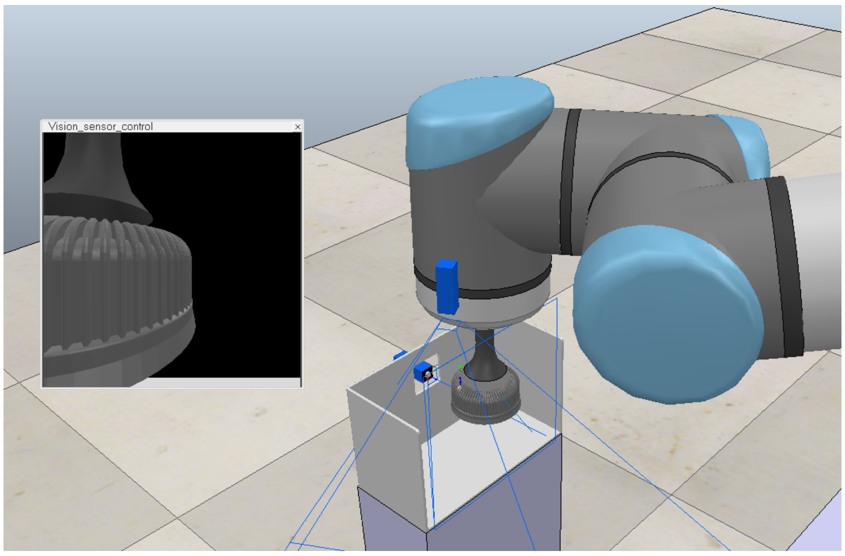


Figure 3.21: Quality control analysis

Placing

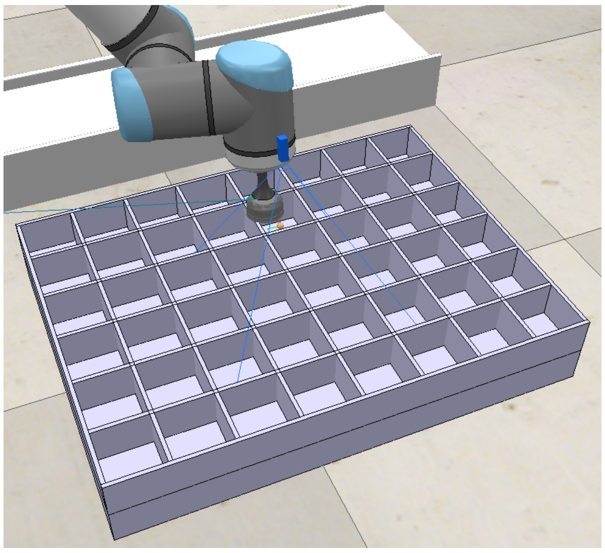


Figure 3.22: Placing phase

In this last simulation part, taking the flowchart as reference, will be explained the placing phase.
As said, after the *ID-reading* phase, the main program flow, knows the ID of the closure that it is processing. This parameter, gets used in this last process step, to allocate in the correct position the closure. Having therefore declared the buffer dimensions: 6 $rows \times 8$ $coloumns$ for 0.455×0.605 $[m]$, it is sufficient to define the buffer central reference frame with respect to the world reference frame, and the simulation computes the goal reference frame to place the closure.
In all those cases in which the operator desires to process the quality control analysis for more than one lap, the placing phase keeps trace, exploiting an array of 48 cells, of the number of processed closures with the same ID.

3.7 Simulation results

The last part of the chapter is structured considering two different scenarios.
In the first one, it is considered a working condition where the operator is never present in the shared workspace. Hence, in this regard what it is under test is not the cycle time but the tool utilized from the robot to attract the closure. In particular, it has been considered the maximum acceleration to which the closure is subjected applying to the end-effector a maximum velocity of 1 m/s and a maximum closure weight of 3.2 g taken from the datasheet. It has then resulted that each gripper was able to apply a sufficient force.
For the second scenario, on the other hand, it has been considered a condition in which the operator is constantly present in the shared workspace. In this case taking as reference that the end-effector is limited to work at maximum 250 mm/s, the analysis parameter is, the cycle time. In order to do this, it has been considered to perform the control analysis for just one closure and to perform only the *thread* quality control position.

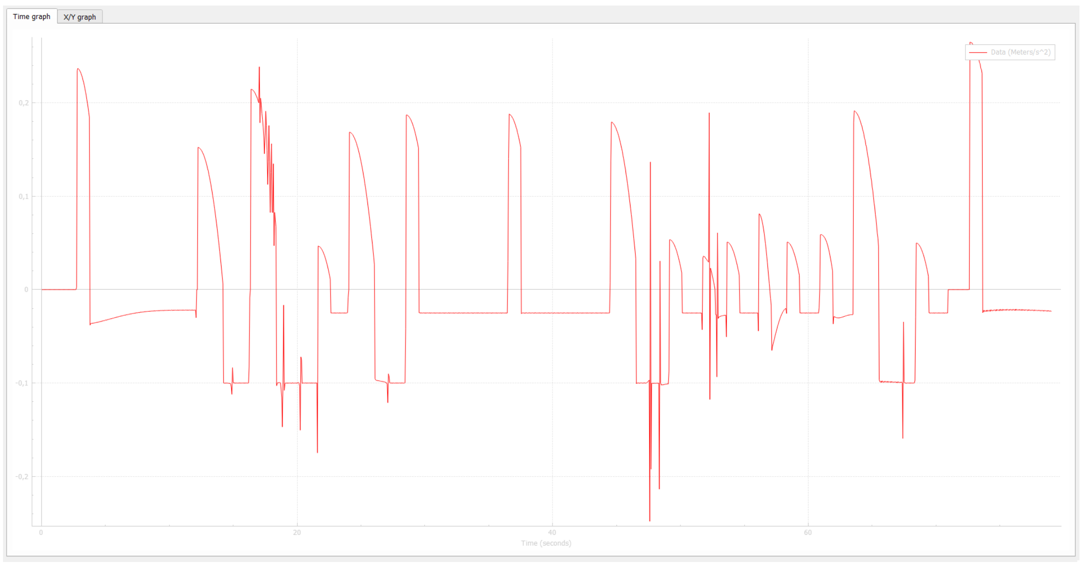


Figure 3.23: End-effector absolute acceleration

It is clearly visible how the simulation is structured following the end-effector absolute acceleration. Even considering the double cycle time to process each control analysis position and considering it as the unit cycle operation time, multiplying this quantity for all the thirteeen analysis positions and adding to each closure a second cutting phase and the placing phase and rounding up each factor, the total cycle time can be computed using the following equation:

$$T_T = ((T_{cu} \times 2) \times N_{cp1l}) + T_{cut} + ((T_{cu} \times 2) \times N_{cp2l}) + T_{pl}$$

where:
T_T is the total time;
T_{cu} is the time unit to complete a single analysis position;
N_{cp1l} is the number of analysis position for the first lap;
T_{cut} is the cutting time;
N_{cp2l} is the number of analysis position for the second lap;
T_{pl} is the placing time.

T_T results to be 4.8 minutes. Rounding up this quantity to 5 mins and multiplying it for all the 48 closures, turns out to have an estimation of the complete quality control operation of $\simeq 240$ mins.

Conclusion

In this company thesis initially it has been studied the problem, analyzing all the characteristics that it composes, trying to maintain all the qualities of the today's process. It therefore has been intoduced a solution, using the collaborative robotic presenting first some basic robotic notions and passing to the main collaborative norms applicable today that dictated the cell design. Successively it has been introduced the simulative environment CoppeliaSim, used to test the chosen manipulator and then proposed three different tool solution that resulted all three suitable to the application. Moreover it has been introduced a new way of closure representation at simulation level through a lower number of parameters. Next the effective simulation comes, declaring first the logical structure using the flowchart and then showing the definitive cell design. Therefore it has been defined the simulation managing through the Grapic User Interface and explaining the comunication method between independent scripts. After that, it has been shown the quality control process through the vision sensor and the products sorting logic into the buffer. Finally some observations on utilized tools an cycle time have been made considering two different scenarios.
Concluding, a large number of future implementations could be provided: the possibility to choose a different number of images for each analysis position, an image acquisition second check given by the operator or a more efficient path planning. All those ideas are just the beginnig, for this reason the source code is downloadable and modifiable.

Bibliography

[1] Alessio Cocchi, *Il futuro è collaborativo. Il vantaggio dei Cobot nei processi di automazione*, `https://blog.universal-robots.com/it/il-futuro-C3A8-collaborativo-cobot-e-automazione`,2019.

[2] SACMI,*Complete plant for the production of plastic caps.*

[3] Claudio Melchiorri, *Elements of Kinematics and Dynamics of Industrial Robots*, Unibo University.

[4] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, Giuseppe Oriolo *Robotics - Modelling, planning and control.*

[5] ISO/TS 15066, *Robots and robotic devices - Collaborative robots*, 2016/02/15.

[6] ISO 10218-1, *Robots for industrial environments - Safety requirements*, 2006/06/01.

[7] ISO 10218-2, *Robots and robotic devices - Safety requirements for industrial robots - Part2*, 2011/04/21.

[8] Rossini, *RObot enhanced SenSing, INtelligence and actuation to Improve productivity and job quality in manufacturing*, 2018/12/31.

[9] Millibar Robotics, *Suction Cups and Suction Pads*, `https://www.millibar.com/product-category/vacuum-gripping-technologies/suction-cups/`.

[10] CoppeliaSim, `https://www.coppeliarobotics.com/`.