POLITECNICO DI TORINO

Master's Degree in MECHATRONIC ENGINEERING

Politecnico
di Torino

Master's Degree Thesis

Development of a simulation model for the verification of a handling system

Supervisors

Prof. Luigi MAZZA

Prof. Carmen VISCONTE

Ing. Giovanni TURCO

Candidate

Cesare GF TAMBURI

July 2022

Summary

The thesis is based on a research work carried out in the system engineering office at SACMI. The main objective of this study is to create a model to verify a handling system. The handling system is a top-loader manipulator called "TLP" in which every automatic machine for the secondary packaging is equipped with. It is a well-known manipulator with two degrees of freedom that can move in a fixed frame. The developed model will be used to verify whether, for a given trajectory to be followed in a given time, the mounted motor can reach the requested torque. This model must be made using the software available in the office. Before creating the model, I analyzed the two main software that the company wants to use: Excel and Solidworks. We found that it is possible to use the Motion of Solidworks to simulate motion and extract torque characteristics. To create the correct model that I wanted to simulate, I simplified the real one, reducing the number of parts, but maintaining the same mass proprieties. This was done to avoid redundancy and unnecessary parts in the assembly. The Solidworks model requires as input the mass properties of the end effector, the path that the manipulator must follow, and the timeline of the trajectory. Firstly, the inverse dynamical problem was solved, which gave as output the angular displacements and velocities of the two motor shafts. Then, with those data, we simulated the direct dynamic and extracted the motor torques. To validate the model developed in Solidworks, I studied the dynamics of each body of the manipulator. The obtained system of equations was then implemented in Matlab and fed with the same data from the Solidworks analysis. The results were compared with the aim of understanding the limitations of the Solidworks model. Finally, both models were compared with real data gathered from a working machine.

Acknowledgements

"The pace of change has never been this fast,
yet it will never be this slow again."
J.T.

Table of Contents

List of Tables

List of Figures

Chapter 1

General Overview

1.1 SACMI

Figure 1.1: SACMI Logo

SACMI "Sociatà Anonima Cooperativa Meccanici Imola" is a metal-mechanic reality founded in Imola in 1919. It produces autonomous machines for ceramic, food and beverage, packaging and plastic. [1]

In 1986 SACMI acquired Inpack, signing their entrance into the production of packing systems for the food industry. Constitute as SACMI Packaging in 2004, it has grown over the years developing machinery for medium velocity with a high technology solution.

Nowadays, SACMI Packaging Chocolate S.p.A. is the only reality in the world that can design and produce machines for chocolate molding, flow-pack packaging and secondary packaging for the food and non-food industry.

Four business units are working inside the company, divided into three operational offices. Wrapping e Tray Forming (Castel San Pietro, BO), Process Moulding (Rozzano, MI), Packaging (Alba, CN).

The topic under study was developed within the business unit for Secondary packaging based in Alba.

1.2 Packaging

The materials in which objects are wrapped before being sold [2]

Very early, food is consumed where found. Families and villages were self-sufficient, making and catching what they used. When containers were needed, nature provided gourds, shells and left them to use. Later, containers were made from natural materials, such as hollowed logs, woven grasses and animal organs.

Fabrics descended from furs used as primitive clothing. Fibers were matted into felts by plaiting or weaving. These fabrics were made into garments, used to wrap products, or formed into bags. The weaving process made grasses and later reeds into baskets to store food surpluses. Some foods could be saved for future meals and less time was needed for collecting food. As discovered of ores and compounds, metals and pottery were developed, leading to other packaging forms. [3]

Nowadays, the packaging also includes marketing, and the definition is more complex. From the modern point of view, the packaging is defined as:

The wrapping material around a consumer item that serves to contain, identify, describe, protect, display, promote and otherwise make the product marketable and keep it clean.[4]

This evolution of packaging was marked in particular by two critical events. First, the advent of the 19th-century universal expositions. Different nations presented their goods no longer in bulk but packages. They collected entire series of industrially manufactured products that were differentiated from each other thanks to the packaging. Furthermore, with the birth of supermarkets and mass distribution outlets. It was essential to distinguish the various articles from those of competitors thanks to the design of ad hoc packaging to ensure the success of a brand.

It carries out two main functions, wrapping the material and promoting it. The package must contain a message that can attract customers to acquire it. It is a design product that must work for marketing.

In the last years, due to climate change and pollution, the packaging industries have moved to become more sustainable and attract more customers. They are introducing new materials to minimize pollution. Biodegradable and recyclable materials aim to create a circular economy to minimize waste and generate new wealth.

1.3 Primary, Secondary and Tertiary Packaging

The packaging can be divided into three main categories: primary, secondary and tertiary.

- **PRIMARY PACKAGING**
  Primary packaging is the packaging directly in contact with the product, referred to as a consumer unit. The main pursuit of primary packaging is to contain, protect and preserve the finished product, especially against contamination.

  The first layer contains the finished product, such as a plastic bag holding cereal or a cardboard box containing a cereals pouch. This type of packaging is planned for the end-user or consumer.

  In addition, it makes it easier for consumers to handle products and makes them look appealing. It can also be used to communicate printed product information to consumers.[5]

Figure 1.2: Wrapping

- **SECONDARY PACKAGING**
  Secondary packaging is the external packaging of the primary packaging. It groups packages and further protects or labels the drug product. Secondary packaging is essential for several reasons, including physical and barrier protection, secondary containment, respect for the regulations, and safety of the customers.

Common types of secondary packaging include cartons and trays. Although secondary packaging can take on various forms, it will always contain a level of primary packaging. Typically, the customer will see at first the secondary packaging; this makes the design of the secondary packaging crucial.

Secondary packaging plays a central role in distribution, presentation, and branding. During distribution, secondary packages shield the primary packages and the product. [6]

Figure 1.3: Secondary Packaging

- **TERTIARY PACKAGING**
  The tertiary packaging aims to group large quantities of secondary containers into a single distribution unit. It makes them convenient to load and unload between warehouses and transport vehicles. The pallet is the world standard for tertiary packaging. Pallets are designed to be easily handled by forklift in warehouse and storage facilities.

  Products in secondary packaging are loaded together onto a pallet at the end of the production. Then they are secured using shrink wrap before loading into a transport vessel, such as a semi-trailer or a shipping container. Pelleted goods are secured and protected from damage during transport. They give the product a minimum of three layers of protection. That makes it more resistant to harsh handling and turbulent transportation conditions. [7]

Figure 1.4: Tertiary Packaging

SACMI works primarily in the primary and secondary packaging leaving the tertiary to other companies. Also, design molding machines for chocolate so it can follow all the processes from cacao bean to a box full of chocolate bars.

On the next page are shown some examples of SACMI machinery. They work in a continuous flow to wrap and pack the products. For the secondary packaging, the machines are divided into three main areas: Forming, Boxing and Closing or Lidding.

Figure 1.5: Primary Packaging - Wrapping

Figure 1.6: Secondary Packaging

.

1.4 Solidworks and Motion Analysis

Before starting the studies for the sizing, the Motion Study of Solidworks is explored to understand possible analyses and which data are extractable from them.

Motion studies are graphical simulations of motion for assembly models. It is possible to incorporate visual properties such as lighting and camera perspective into a motion study. Motion studies do not change an assembly model or its properties: they simulate and animate the motion defined for a model. SOLIDWORKS mates restrict the motion of components in an assembly when the model moves.

During the thesis, Motion Analysis is used to accurately simulate and analyze the effects of motion elements (including forces, springs, dampers, and friction) on an assembly. Motion Analysis uses computationally strong kinematic solvers. The analysis accounts for material properties, mass, and inertia. It can also be used to plot simulation results for further analysis.

Next, a brief introduction to the different aspects of Solidworks used to create the simulation is reported. It is helpful to understand why different simplifications were made in the following chapters.

All the definitions are taken from the Solidworks Online Help. [8]

1.4.1 Hinge mates

A hinge mate limits the movement between two components to one rotational degree of freedom. It has the same effect as adding a concentric mate plus a coincident mate.
For Concentric Selections, select two valid entities that will remain concentric during the movements. For Coincident Selections, select two proper planes or planar faces that will remain on the same plane during the movement.
It is also possible to specify angle limits, but it is preferred to use a specific mate to modify it easier.

1.4.2 Limits angles

Limit mates allow components to move within a range of values for distance and angle mates. Specify a starting angle with a maximum and minimum value.

To easily understand the position and limits of the angles, the plane of construction is used to set their limits. The values are taken from the drawing of the TLP.

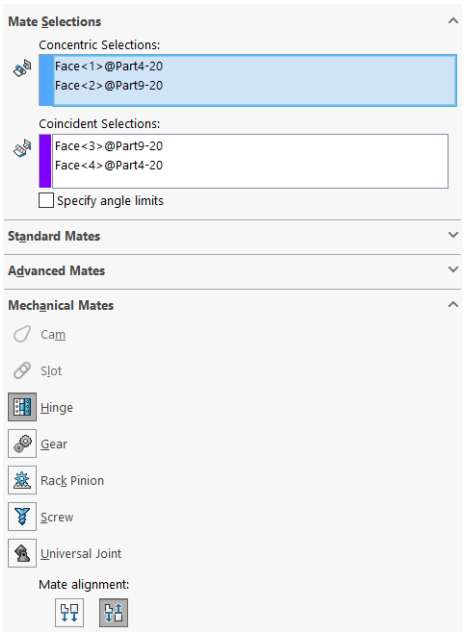

Figure 1.7: Hinge Mate setting

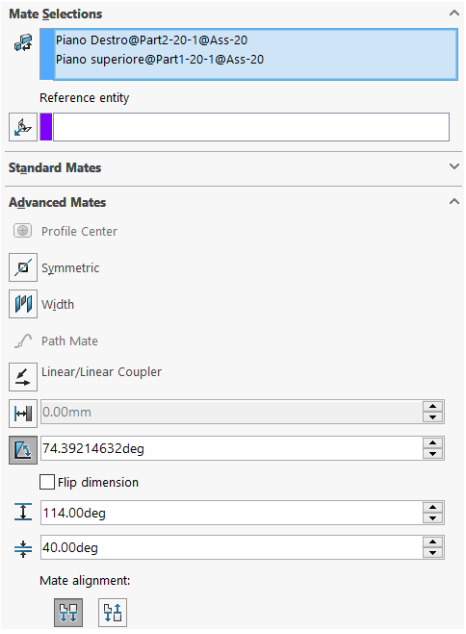

Figure 1.8: Limit Angle setting

.

1.4.3 Path mate

A Path mate constrains a selected point of a component to a path. The path is defined by selecting one or more entities in the assembly and composing a Spline with the command *Fit Spline*. It is possible to define the component's pitch, yaw, and roll as it travels along the path. This particular mate is indispensable to creating a Motor in the simulation.

The path is a part of the assembly and matches a specific position on the Front Plane. Then all the path components are **Fixed** to lock its movements. The selected point will follow the trajectory defined by the path.

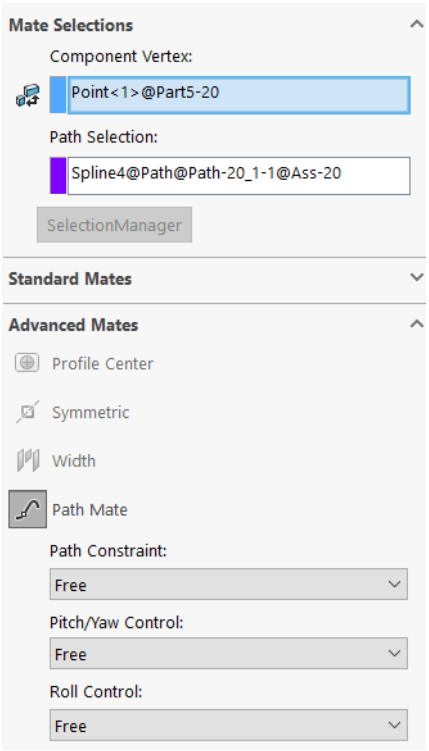

Figure 1.9: Path Mate setting

The Path Mate has some limitations because the spline that forms it is not optimized. In the simulation, the spline is created from a polygon. When interpolated by the spline, the vertices of this polygon create some discontinuity that is not easy to follow by the Path Mate motor.
Different errors are produced in those parts. They will be analyzed in detail later.

1.4.4 Motion interface

Before continuing to the proprieties of the Motion, the add-ins *SOLIDWORKS Motion* must be activated. The license in use, both at Politecnico and SACMI, permits this analysis.

The interfaces found are reported in the following image.

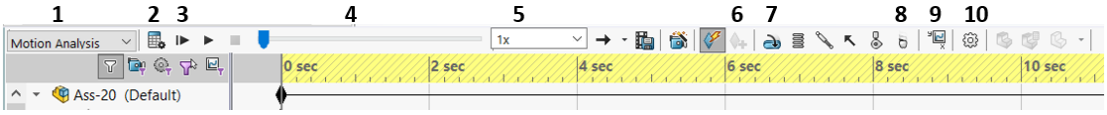

Figure 1.10: Motion Analysis Interface

Every component has a specific meaning and use. For someone, the details are reported in the next section.

1. **Motion Analysis** I have similar interface for *Automation, Basic Motion and Motion Analysis* but I must stay in this configuration to have the possibility to create motor and extract the results.
2. **Calculate** To calculates the Motion Study.
3. **Play From Start**
4. **Timeline** Move into the execution of the motion
5. **Playback Speed** Useful to set the playback speed multiplier or the total duration. By default is set at 1x or 5sec, those data are needed for the Study Proprieties.
6. **Add/Update Key** Used to Create a new key with the selected item's current attributed (position velocity), or updates an existing key.
7. **Motors** Moves a component as if acted upon by a motor.
8. **Gravity** Adds gravity to the study.
9. **Results and Plots** Calculates results and create graphs.
10. **Motion Study Proprieties** Specifies simulation proprieties for the Motion Study.

1.4.5 Motion study proprieties

Before creating a motor and simulating the motion, the meaning of the parameters of the Motion Study Proprieties must be understood. In the simulation, those parameters are changed different times to have a better result.

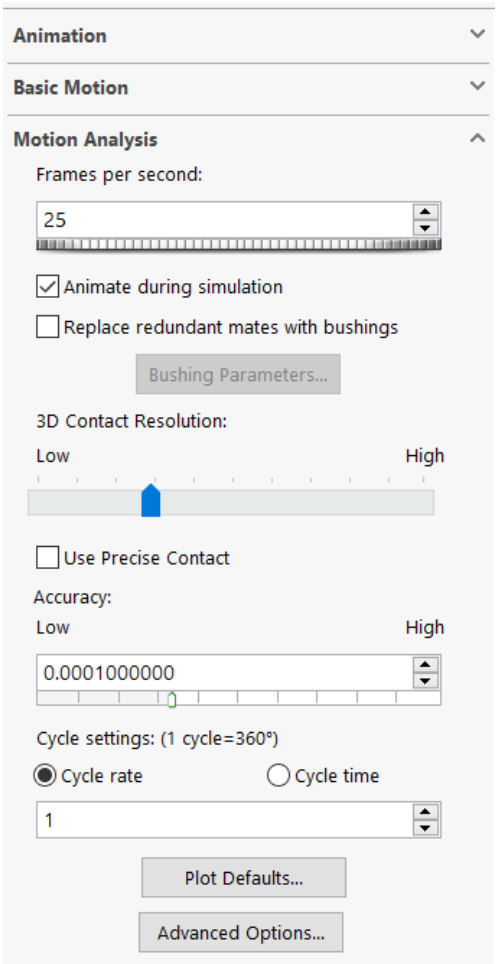

Figure 1.11: Motion Study Default Proprieties

- **Frames per second** This value, multiplied by the length of the animation, specifies the total number of captured frames. This value does not affect the playback speed. By default, it is set to 25 for 5 seconds of simulation, so it will take five frames per second. Increasingly it is possible to collect more value from the data during the simulation.

- **Animate during simulation** Clearing this option speeds up the calculation time and prevents the graphics from displaying the motion during the simulation calculation. Useful for time-consuming calculation.
- **Accuracy** Higher values increase the required calculation time.
- **Cycle settings** Specifies the cycle rate or period. Cycle settings define the cycle angle used in custom Motor or Force profiles. Cycle time is selected and specify the cycle period in seconds, decreasing its default value to 0.2 sec.

Other Proprieties are not used in the project analysis.

1.4.6 Motor

Motors are motion study elements that move components in an assembly by simulating the effects of various types of motors. They can be set from the motor Manager toolbar.

There are three different motor types *Rotary motor, Linear motor and Path Mate motor*. The last specifies, for a selected path mate in the assembly, displacement, velocity, or acceleration as a body moves along a path. It is the first used in the project analysis.

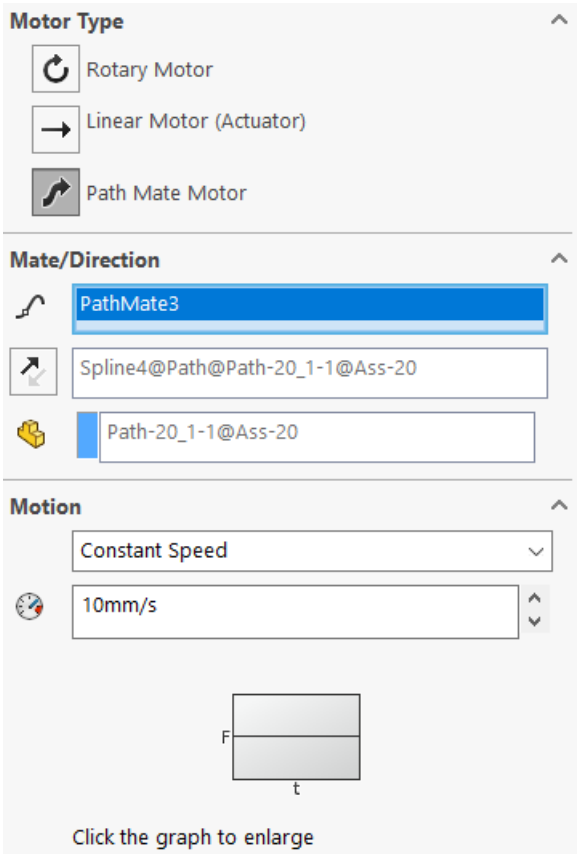

Figure 1.12: Path Mate Motor Setting

To specify the propriety of the Path Mate Motor, the path mate must first be selected. It defines the path of the motion. After defining the path, this passage can be quickly done from the Feature Manager Design tree.

The direction of the motion can be reversed using the following button with two arrows. If the path component is fixed as done previously, the component relative to which the motion is applied is automatically selected.

Next is time to define the motion. As shown in the figure, different motion functions can be chosen: constant speed, distance, oscillating, segments, data points, expression, and servo motor.

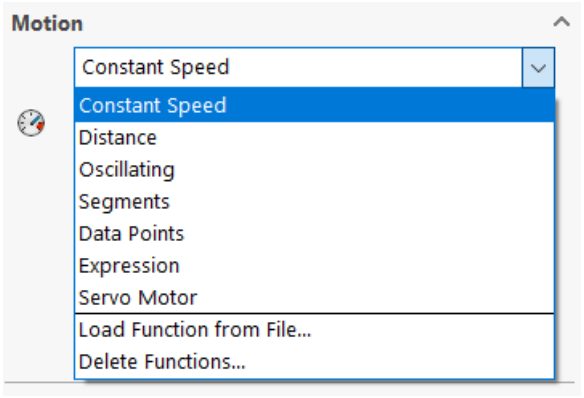

Figure 1.13: Motor Profile Selection

In our analysis, different types are used depending on the accuracy needed and the definition of the data collected.

The first analysis uses the constant velocity obtained from the cycle time.

1.4.7 Results

The position to collect data and the type of data needed must be defined to export data from the simulation. From the result setting is possible to define the Category, Sub-Category and Component (X, Y, Z, magnitude) for the resulting graph. Next, it must define the Entities to measure.

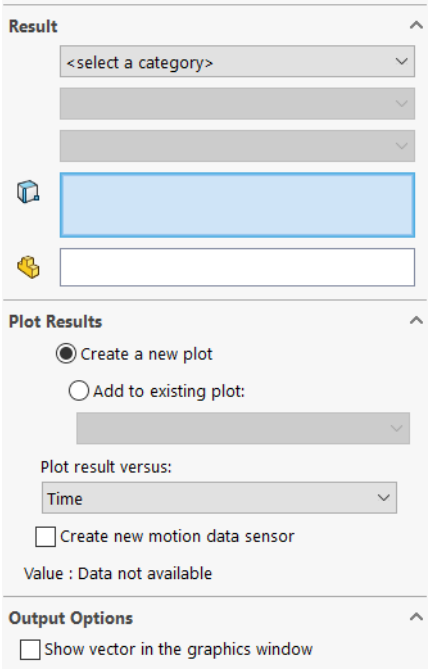

Figure 1.14: Results setting

Different Categories and Sub-Categories are possible to select, but they are available only for some types of Entities. The categories are analyzed and divided into entities from which it is possible to collect information. The four entities that are possible to measure are Shaft, Motor, Mates and Contact.

The following four tables report all the Categories and the measurable Entities. The category components that can be measured are not reported, but they are the X, Y, Z, the magnitude and the normal and tangential components for some categories.

The plots obtained can be modified from the chart properties. The data can be exported in a CSV file for analysis in other programs.

Displacement/Velocity/Acceleration	
Trace Path	**SHAFT**
Center of mass position	**SHAFT**
Linear Displacement	**SHAFT**
Linear Velocity	**SHAFT**
Linear Acceleration	**SHAFT**
Angular Displacement	**SHAFT**
Angular Velocity	**SHAFT**
Angular Acceleration	**SHAFT**

Table 1.1: Displ/Vel/Acc

Forces	
Motor Force	**MOTOR**
Motor Torque	**MOTOR**
Reaction Force	**MATES**
Reaction Moment	**MATES**
Friction Force	**MATES**
Friction Moment	**MATES**
Contact Force	**CONTACT**
Angular Acceleration	**SHAFT**

Table 1.2: Forces

Momentum/Energy/Power	
Translational Momentum	**SHAFT**
Angular Momentum	**SHAFT**
Translational Kinetic Energy	**SHAFT**
Angular Kinetic Energy	**SHAFT**
Total Kinteic Energy	**SHAFT**
Potential Energy Delta	**SHAFT**
Power Consmption	**MOTOR**
Angular Acceleration	**SHAFT**

Table 1.3: Momentum/Energy/Power

Other Quantities	
Euler Angles	**SHAFT**
Pitch/Taw/Roll	**SHAFT**
Rodriguez Parameters	**SHAFT**
Bryant Angles	**SHAFT**
Projection Angles	**SHAFT**
Reflected Load Mass	**SHAFT**
Reflected Load Inertia	**MOTOR**

Table 1.4: Other Quantities

1.5 Initial consideration for motion analysis

After studying the Motion environment, some concerns can be made about structuring the thesis work. Some conclusions are taken from the simulations that are explained later in the second chapter.

- The **Fit Spline** command can create path starting from an open polygon that represent the real path to simulate. (Figure 1.15)
- To Follow exactly the path during simulation, the edge must be smoothed. It is possible to do this in 2 ways. One is smoothing the polygon and then creating the spline on it. The second is increasing the Tolerance in the Spline proprieties that automatically generate the smoothness in the spline. (Figure 1.16)
- A tip learned is to leave open the polygon in all the places where there is a back and forth movement. In that way, the spline creates a better path that the simulation follows more easily.
- To decide the initial position of the motion, the Timeline must first reset to the initial position. Then pressing the **Add/Update Key** I will set this position as the initial ones.
- To modify the sample time obtained in the Results and Graph, the motion study properties must be modified. First, it is recommended to increase the frame per second up to 100, then deselect the "Animate during simulation" boxes and try the simulation. Second, create a trace path graph, if it is loyal to the theoretical ones, the frames per second are enough, so it is possible to read the result. If not, it is better to increase the frame per second. Higher numbers mean a longer time for the simulation.
- When we are satisfied with the path the manipulator follows, it is time to extract the data from the simulation. The torque cannot be directly extracted from the path mate simulation. Some steps must be taken to obtain the final torque. First, a path simulation is performed, the angular position is extracted from the motor shaft, new motion analysis is performed, a motor with the same performance extracted from the first simulation is created, the simulation is repeated, and the torque is extracted.

That was the work done during the first two weeks of the internship.

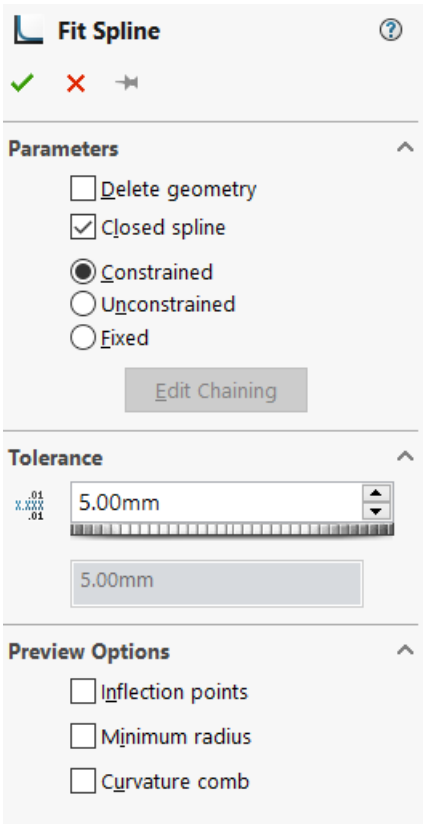

Figure 1.15: Spline setting

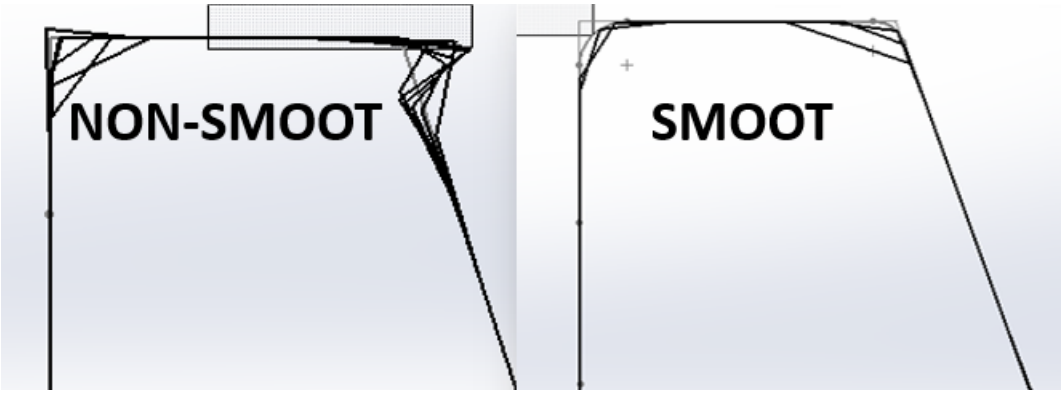

Figure 1.16: Smoothed Spline

Chapter 2

TLP - Top Loader Performance Manipulator

2.1 Description

Top Loader Performance manipulators are the most used manipulators in secondary packaging machines in SACMI. It is known as TLP.

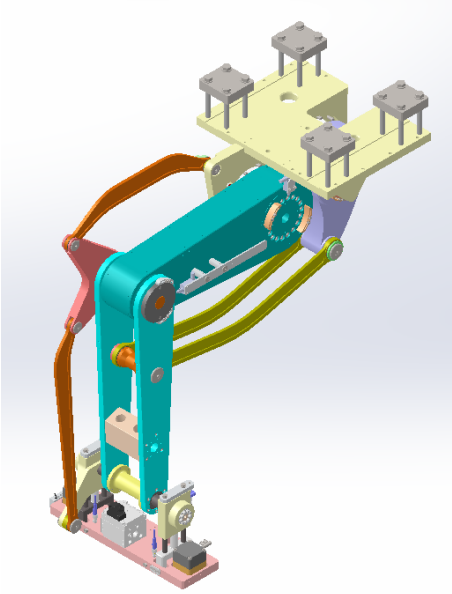

Figure 2.1: TLP

TLP is a manipulator with two degrees of freedom; it can move in a fixed plane on the vertical and horizontal axes. The end effector has a kinematic that fixes its orientation: it is maintained parallel to the horizontal plane. At this horizontal plane is attached the end effector that is used for the needed task. If needed, additional degrees of freedom are created in the end effector. In the next chapter, we consider the end effector.

TLP is a manipulator that has been used for many years, created within SACMI from another similar manipulator called TLA. During the years, it evolved not from an engineering study but experience and practical testing. Also, the performances of its movements are collected from different applications. The thesis aims to create a manipulator model that can be used to study its performance on a given path.

Figure 2.2: TLA

The TLP is moved by two different motors and reducers that are independently controlled. The reducers are manufactured by SPINEA, while the motors are purchased from SCHNEIDER or ROCKWELL. The motors are connected to a BUS that powers different components. The BUS voltage is set at 500V. There is no information on how the voltage is divided between the components. In addition, each motor is controlled by a single drive that modulates the current.

Figure 2.3: TLP Motors

The principal work of the manipulator is to move products or boxes from one position to another. It also forms cardboard into boxes after they are taken from a feeder. The trajectory follower is simplified as possible: straight lines, up-down, connected directly by the controller's software. The setting of the path is done through an HMI "human-machine interface". Here is possible to see the characteristic points where the TLP must pass.

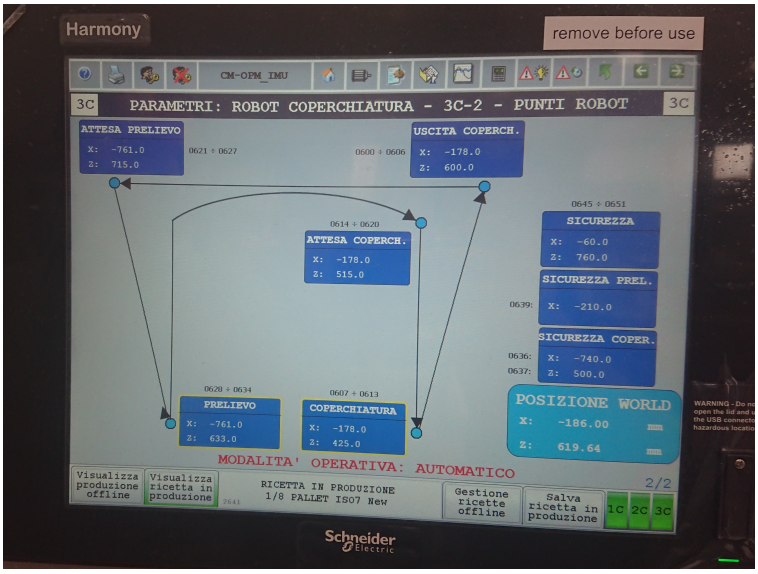

Figure 2.4: HMI

The numbers of points are directly proportional to the complexity of the trajectory; typically, there are at least four points. In the testing phase, the programmer can modify the position of those points while the controller creates the trajectory directly.

We must remember that the TLP is **not** a studied "engineered" manipulator, is a working, efficient, and cost-effective solution.

2.2 3D Model and simplifications

Before creating the model, the TLP from an assembly in Solidworks is analyzed. It incorporates nine principal sub-assemblies, referred to as Body, plus the fixed base.

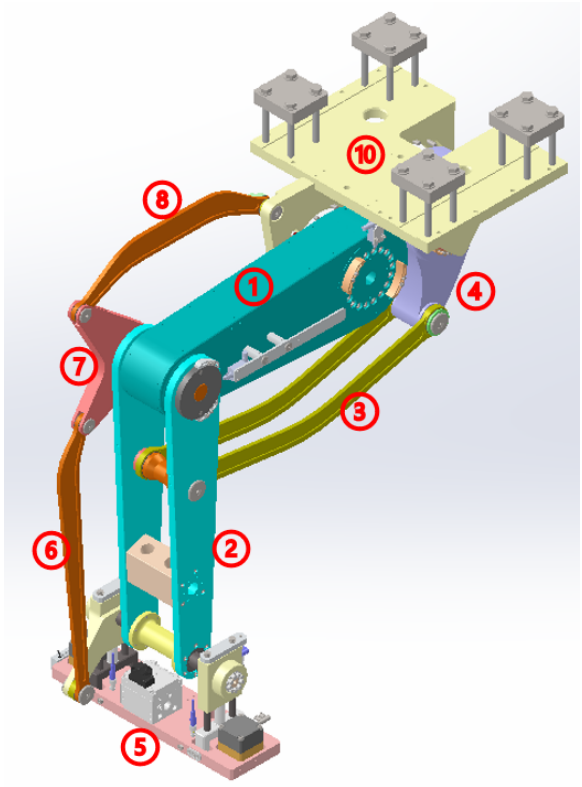

Figure 2.5: TLP Numerated

Figure 2.6: TLP Sub-Assembly numbers

In the assembly of a complete TLP, more than three hundred single components mated together are needed. This high number of components does not allow us to simulate the motion using the original design directly. The number of single components and the number of faces are reduced to allow a simulation using the available calculators.

While redesigning the nine sub-assembly, the mass properties are conserved unaltered. To quickly redesign, there is a function of Solidworks that permits us to override the Mass Properties of every component.

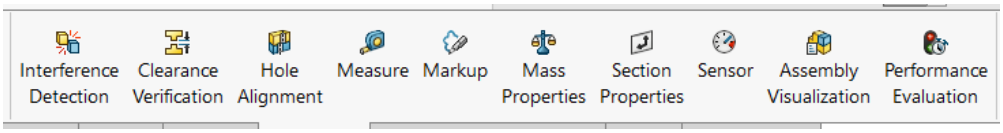

Figure 2.7: Mass Properties

The principal properties of a component are the following:

- Mass **Kg**
- Volume **Cubic Meters**
- Surface Area **Square Millimeters**
- Center of Mass **Millimeters**
- Axes and Moments of inertia at the center of mass **Kilogram * Square Millimeters**
- Moments of inertia aligned with the output coordinate system

The simplified TLP assembly is reported in the following figure. To simplify the model and work only on a 2D plane, only the X and Y center of mass variation and the moment of inertia along the Z axes are considered.

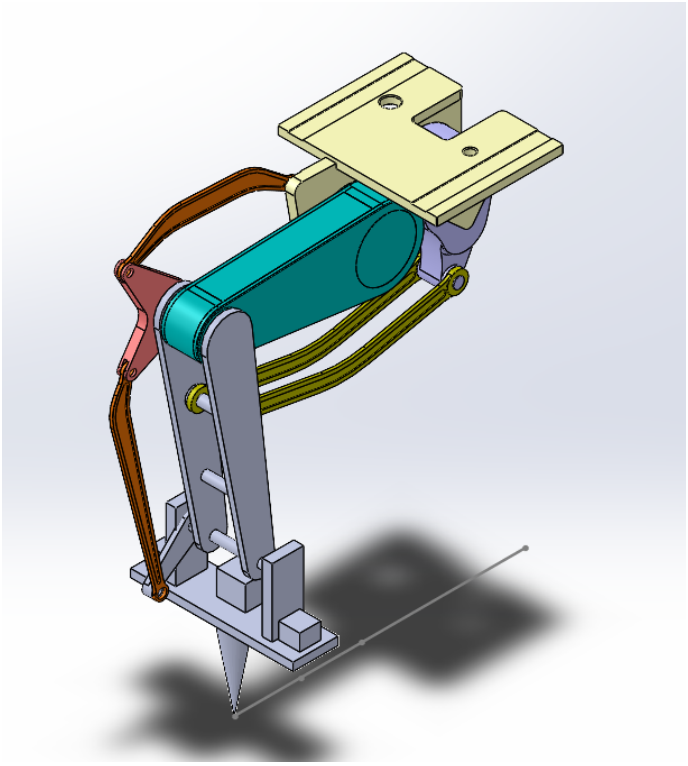

Figure 2.8: Simplified TLP

Only the distance from the component's origin to the center of mass is needed. They are the same data used in Matlab. The origin of each component is based on the first shaft, the closest to the motor.

For body 3, to simplify the Matlab model, the mass propriety is duplicated to have a single body move in the Matlab model. The Solidworks model has two components, but the result does not change.

On the next page, all the mass information for each original body is reported. Measurements are evaluated in **Kg** and **mm**. Therefore, the momentum of inertia is expressed in $Kg \cdot mm^2$

	Mass	**Center of Mass**	**Momentum of Inertia**
Body	**Kg**	**mm**	$Kg \cdot mm^2$
1	16.56	224.39	866 843
2	17.77	253.89	1 057 377
3	5.13	376.18	345 814
4	10.93	93.38	157 399
5	16.26	97.73	103 738
6	1.854	330.47	81 591
7	2.79	72.04	25 573
8	1.739	280.86	47 749

Table 2.1: Mass Propriety

After some tests, we understand that overwriting the momentum of inertia can create problems. Only the Mass and the Center of Mass are changed.

2.3 Working Area

The TLP works in a 2D plane. The position of the end effector is controlled by the motion of the two motors **A** **B**, two independent angles give us the X and Y positions of the end effector.

Before studying dynamic theory, the EE working area and the relationship between angles and EE positions are analyzed. First, every joint and angle are named. Also, each angle reference system is set. They are described in the figure below.

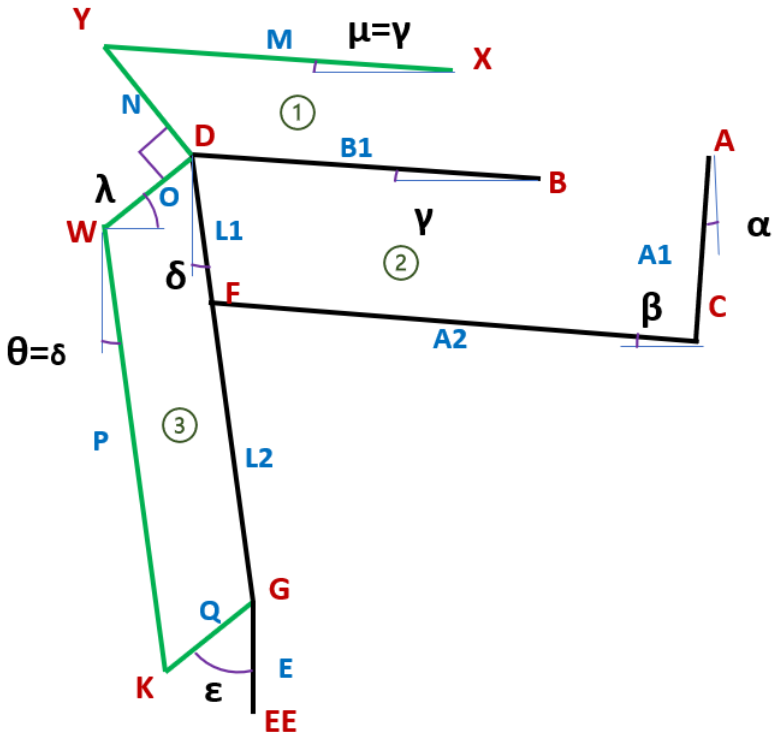

Figure 2.9: Joint TLP model

A standard reference system is defined for a 2D plane with origin below the TLP. The reference system is set up from a known working area and set the origin that can be seen in Figure 2.10.

After some study, the reference system is different to the one used in production, but the two are compatible and can be correlated with translation and rotation. So, all analyses are done with the defined reference system.

Then at each joint is assigned the position with respect to the end effector's position. They are reported in the table below.

	X	**Y**
A	$C_x - A_1 sin(\alpha)$	$C_y + A_1 cos(\alpha)$
B	$D_x + B_1 cos(\gamma)$	$D_y - B_1 sin(\gamma)$
C	$F_x + A_2 cos(\beta)$	$F_y - A_2 sin(\beta)$
D	$F_x - L_1 sin(\delta)$	$F_y - L_1 cos(\delta)$
E	E_x	E_y
F	$G_x - L_2 sin(\delta)$	$G_y + L_2 cos(\delta)$
G	E_x	$E_y + E$
X	X_x	X_y
Y	$D_x - N sin(\lambda)$	$D_y + N cos(\lambda)$
W	$K_x - P cos(\theta)$	$K_y - P sin(\theta)$
K	$E_x + \Delta X$	$E_y + \Delta Y$

Table 2.2: Joint Position

I can do some consideration on the angles.

- Taking into account the first parallelogram (Figure 2.9, green circle), it is possible to see that $\mu = \lambda$ in fact I have $\mathbf{XY = DB}$ and $\mathbf{YD = BX}$.
- The same can be sad on the third parallelogram. $\theta = \delta$ because $\mathbf{DG = KW}$ and $\mathbf{WD = GK}$.

Furthermore, two angles remain fixed in all movements:

- ϵ because it is a constructive angle.
- λ stays fixed at 40 degrees for the kinematic. The green chain is used only to keep the end effector plane in the horizontal position.

To determine the value of each angle, a system is written to relate the end effector's known position to the fixed joint: **A and B**. The values of the fixed joint referred to the reference system shown in Figure 2.10 are reported in the following table.

	x	y
A	1096.2	1283.4
B	850	1240
X	702.2	1416.2

Table 2.3: Fixed Position

The system of equations is reported below.

$$A_x = E_x - L_2 sin(\delta) + A_2 cos(\beta) - A_1 sin() \tag{2.1}$$
$$A_y = E_y + E + L_2 cos(\delta) - A_2 cos(\beta) + A_1 cos() \tag{2.2}$$
$$B_x = E_x - L_2 sin(\delta) - L_1 sin(\delta) + B_1 cos(\gamma) \tag{2.3}$$
$$B_y = E_y + E + L_2 cos(\delta) + L_1 cos(\delta) - B_1 sin(\gamma) \tag{2.4}$$

It is a linear system of four trigonometric equations in four unknowns. It is solvable. It only has to understand if it is solvable algebraically or numerically. After some attempts with the parametric approach, it is decided to solve numerically using Matlab with the solver *fsolve*.

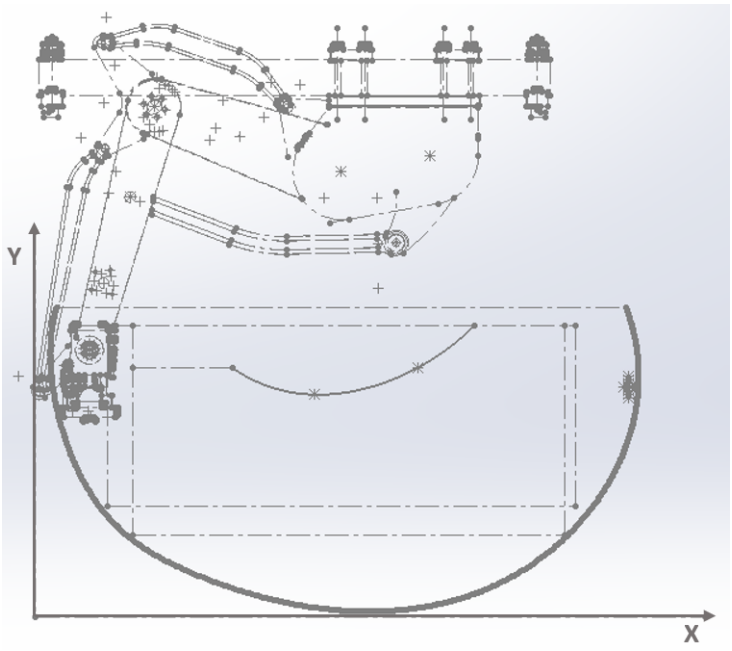

Figure 2.10: Pre-Study working area and System

The program is created with a function that inputs the end effector's position and gives the value of the four angles in radiant as an output. To study all theoretically possible positions, two chained **FOR** cycles are created, one for the reachable X and the second for the reachable Y.

Their limits are $X \in [0, 1675]$ and $Y \in [0, 860]$, they are defined from the dimension of the half moon in Figure 2.10.

The main program refers to a function that defines the four equations and all known dimensions. They are reported in *Appendix A*

With all the angles available, it is to understand how the work area is defined. The angle's mechanical limits imposed on the **A, B, D** joints are analyzed. It is possible to see the limits mechanism highlighted in the figure below.

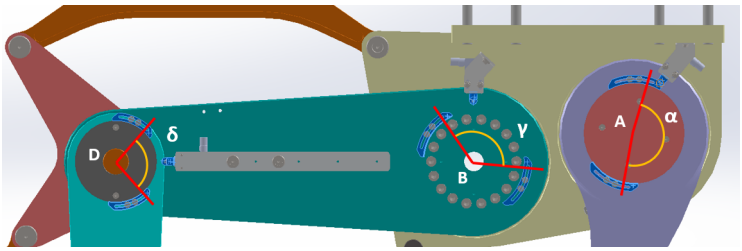

Figure 2.11: Limit Angle

The limitation mechanism is for safety and backup because the angles are mainly controlled by the PLC that activates the motors. Also, the impenetrability of the body must be considered. Solidworks does not directly control this aspect, so the limited angles are assigned to respect these conditions.

The limits assigned are reported in the following table. They are taken from different reference systems, α from the horizontal axes, γ from the vertical axes, and the last from the right plane of body one.

	min	**MAX**
α	-35	150
γ	-110	40
$\delta + \gamma$	-24	50

Table 2.4: Angles limits

The true working area is confronted with the known one. The comparison is made only with acceptable angles plotted using the function *contour*. This function draws a contour plot of the Z matrix in the XY plane. The program used to plot the area is reported in *Appendix B*. The reachable area is blue, while the inaccessible area is yellow.

The two areas differ in some points, especially in the upper part. However, this does not interfere with the final performance characteristic. It is because the vertical position of the TLP is manually adjusted during the assembly of the whole machine. So, if the operators see that the TLP has difficulty reaching some points, the easiest solution is to translate the TLP vertically.

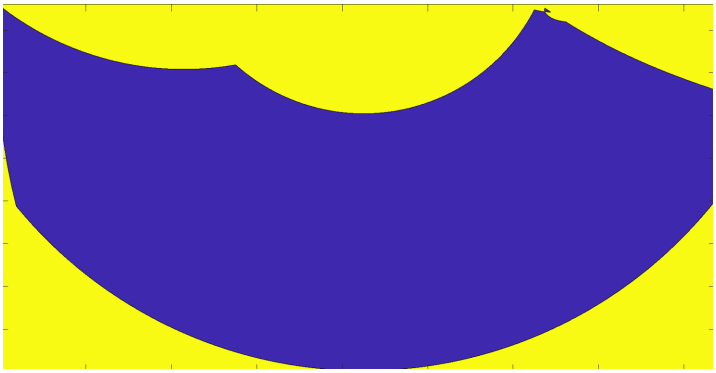

Figure 2.12: Working Area

In the study of the dynamics, a different configuration for the angle is used. This is because every angle position, velocity, and acceleration must be studied. So a new reference system is created , it is easy to connect to the original ones:

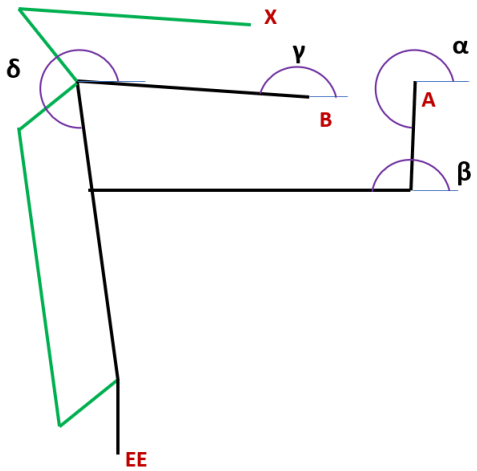

Figure 2.13: Dynamic angle

2.4 Motor and Reducer

ROCKWELL and SCHNEIDER manufacture the main electrical motors used in SACMI for TLP. In particular, the two models used in the TLP are as follows:

- SCHNEIDER SH140/30/120
- AB ROCKWELL

In both cases, they are three-phase motors powered by an alternate current at 400V. Next, the datasheets for the SCHNEIDER motor are reported. It is the one used in the thesis.

Technical Data

Winding data for 3 phase AC 400 V

Reference Data	Abbrev	Unit	SH-140 30 120
Standstill torque	M_0	Nm	11.4
Rating speed	n_N	min^{-1}	3000
Peak torque	M_{max}	Nm	26.0
Rotor moment of inertia	J_M	$kgcm^2$	7.41
Weight	m	kg	11.9
Number polepairs	p		5
Voltage constant Ph-Ph (120°C)	K_E	$V/kmin^{-1}$	97
Standstill current	I_0	A	7.8
Maximum current	I_{max}	A	21.6
Rated torque	M_N	Nm	8.4
Rated power	P_N	kW	2.64
Rated current	I_N	A_{rms}	5.7
Torque constant (120°C)	k_T	Nm/A_{rms}	1.46
Winding resistance Ph-Ph (20°C)	R_{U-V}	Ω	1.3
Winding inductance Ph-Ph	L_{U-V}	mH	14.7

Figure 2.14: SCHNEIDER Motor

SPINEA manufactures the main reducer used by SCAMI in TLP, mainly a T-200 series reducer with a reduction ratio of $i = 83$.

Figure 2.15: SPINEA reducer

The T series is chosen for its high-precision reduction gears with a cylinder-shaped case. They comprise an accurate reduction mechanism and high-capacity radial and axial cylindrical roller bearings. This design of reduction gears allows for the mounting of the load directly on the output flange of the motor integrated directly into the main base of the TLP.

The main technical characteristics of the reducer **TS-200 83** are reported in the following pages.

Tab. 2.3c: T series rating table

Size	Reduction ratio	Rated output torque	Max. acceleration / deceleration torque	Permissible output torque at emergency stop	Rated input speed	Max. allowable input speed 9)	Tilting stiffness 1) 5)	Torsional stiffness 1) 6)	Max. no-load starting torque 8)	Max. back driving torque 8)
	i	T_R [Nm]	T_{acc} [Nm]	T_{em} [Nm]	n_R [rpm]	n_{max} [rpm]	M_t [Nm/arcmin]	k_t [Nm/arcmin]	[Nm]	[Nm]
TS 200	**63**	890	2 225	4 450	2 000	3 500	1 070	178	1.90	90
	83					4 000			1.80	120
	125					4 000			1.70	200
	169					4 500			0.90	210

Figure 2.16: TS-200 datasheets

Tab. 2.3c: T series rating table - continued

Size	Reduction ratio	Max. lost motion	Average angular transmission error 1) 6)	Hysteresis	Max. moment 2) 3)	Rated radial force 2)	Max. axial force 2) 4)	Input inertia 7)	Weight 7)
	i	LM [arcmin]	ATE [arcsec]	H [arcmin]	$M_{c\,max}$ [Nm]	F_{rR} [kN]	$F_{a\,max}$ [kN]	I [10^{-4} kgm²]	m [kg]
TS 200	**63**	<1.0	±18	<1.0	3 300	21.1	31.7	2.6	17.23
	83								
	125								
	169								

Figure 2.17: TS-200 datasheets

.

2.5 Dynamics

Studied the kinematics, the free-body diagram for each TLP component is written. Bodies are considered inextensible with a concentrated mass.

Angle, angular velocity, and acceleration are taken from the horizontal axes in a positive counterclockwise direction according to the reference system sown in Figure 2.18.

The orientation system used for all studies is described below. The horizontal and vertical equations are written according to the direction shown. The same is also true for the torque equation in counterclockwise verse.

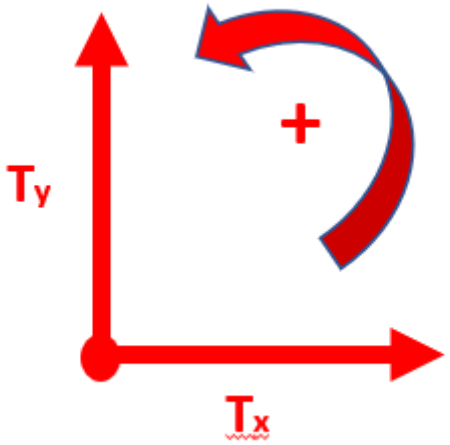

Figure 2.18: Orientation System

2.5.1 Body 1

It is the main arm of the TLP, named $B\bar{D}$, and has a torque applied to shaft B by the reducer. The torque applied is C_b and the angle at this point is γ with its corresponding velocity and acceleration. The constraining forces are not an interest of the thesis, so their value is not calculated. It is in common with the first and third parallelograms.

To better define angle, angular velocity, and acceleration, I will use the following definition:

$$\gamma, \quad \dot{\gamma}, \quad \ddot{\gamma}$$

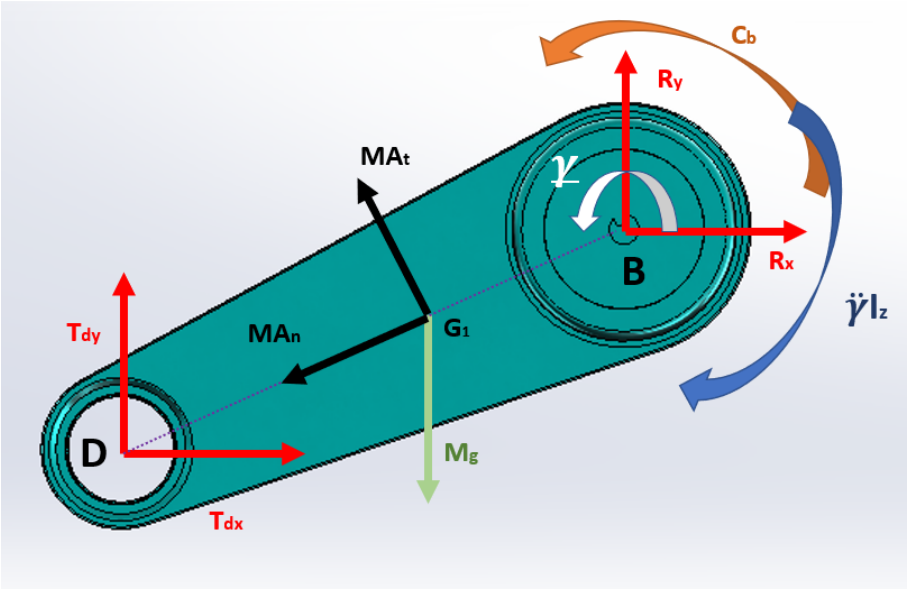

Figure 2.19: DCL Body 1

$$T_{DX} + R_X - M_1 \cdot \dot{\gamma}^2 \cdot B\bar{G}_1 \cdot cos(\gamma - 180) - M_1 \cdot \ddot{\gamma} \cdot B\bar{G}_1 \cdot sin(\gamma - 180) = 0 \tag{2.5}$$

$$T_{DY} + R_Y - M_1 \cdot \dot{\gamma}^2 \cdot B\bar{G}_1 \cdot sin(\gamma - 180) + M_1 \cdot \ddot{\gamma} \cdot B\bar{G}_1 \cdot cos(\gamma - 180) - M_1 g = 0 \tag{2.6}$$

$$\begin{aligned} C_b - T_{DY} \cdot B\bar{D} \cdot cos(\gamma - 180) + T_{Dx} \cdot B\bar{D} \cdot sin(\gamma - 180) + M_1 \cdot g \\ + B\bar{G}_1 \cdot cos(\gamma - 180) - \cdot M_1 \cdot \ddot{\gamma} \cdot B\bar{G}_1{}^2 - \ddot{\gamma} \cdot I_{Z_1} = 0 \end{aligned} \tag{2.7}$$

2.5.2 Body 2

It is the vertical arm of the TLP, has different connections to other arms, and forms the second parallelogram.
The angle, angular velocity, and acceleration are define as:

$$\delta, \quad \dot{\delta}, \quad \ddot{\delta}$$

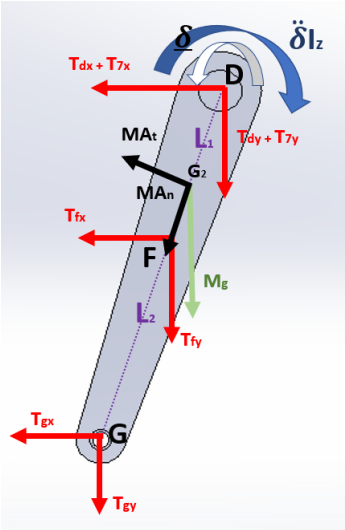

Figure 2.20: DCL Body 2

$$-T_{DX} - T_{7X} - T_{FX} - T_{GX} - M_2 \cdot \ddot{\delta} \cdot D\bar{G}_2 \cdot cos(\delta - 180) - M_2 \cdot \dot{\delta}^2 \cdot D\bar{G}_2 \cdot sin(\delta - 180) = 0 \tag{2.8}$$

$$-T_{DY} - T_{7Y} - T_{FY} - T_{GY} + M_2 \cdot \ddot{\delta} \cdot D\bar{G}_2 \cdot sin(\delta - 180) - M_2 \cdot \dot{\delta}^2 \cdot D\bar{G}_2 \cdot cos(\delta - 180) - M_2 \cdot g = 0 \tag{2.9}$$

$$+T_{GY} \cdot \bar{L} \cdot cos(\delta - 180) - T_{GX} \cdot \bar{L} \cdot sin(\delta - 180) + T_{FY} \cdot \bar{L_1} \cdot cos(\delta - 180)$$
$$-T_{FX} \cdot \bar{L_1} \cdot sin(\delta - 180) + M_2 \cdot g \cdot D\bar{G}_2 \cdot cos(\delta - 180) - M_2 \cdot \ddot{\delta} \cdot D\bar{G}_2{}^2 - I_{Z_2} \cdot \ddot{\delta} = 0 \tag{2.10}$$

2.5.3 Body 3

It transfers the motion from body 4 to the middle joint of body 2. A single body is used to analyze the dynamics instead of two, the mass and the inertia are doubled. Displacement, angular velocity, and acceleration are named as follows.

$$\beta, \quad \dot{\beta}, \quad \ddot{\beta}$$

The ℵ "Alaph" angle is defined as:

$$\aleph = \angle FCG - (180 - \beta)$$

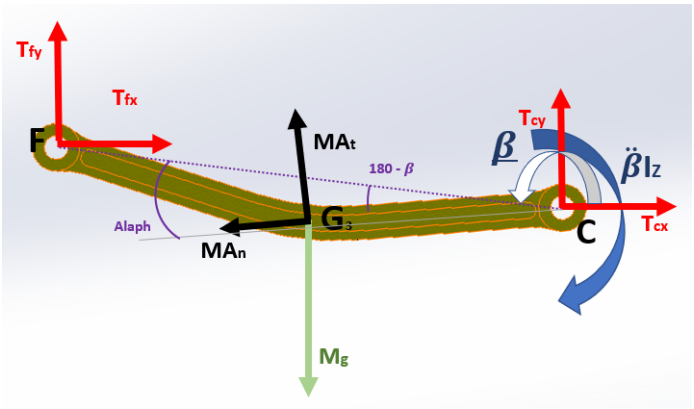

Figure 2.21: DCL Body 3

$$T_{FX} + T_{CX} - M_3 \cdot \ddot{\beta} \cdot C\bar{G}_3(\aleph) - M_3 \cdot \dot{\beta}^2 \cdot C\bar{G}_3 \cdot cos(\aleph) = 0 \tag{2.11}$$

$$T_{FY} + T_{CY} + M_3 \cdot \ddot{\beta} \cdot C\bar{G}_3(\aleph) - M_3 \cdot \dot{\beta}^2 \cdot C\bar{G}_3 \cdot sin(\aleph) - M_3 \cdot g = 0 \tag{2.12}$$

$$\begin{aligned} -T_{FY} \cdot C\bar{F}(180 - \beta) - T_{FX} \cdot C\bar{F}(180 - \beta) + M_3 \cdot g \cdot C\bar{G}_3 \cdot cos(\aleph) \\ -M_3 \cdot \ddot{\beta} \cdot C\bar{G}_3{}^2 - I_{Z_3} \cdot \ddot{\beta} = 0 \end{aligned} \tag{2.13}$$

2.5.4 Body 4

Here, the second torque C_a is applied. The reducer used is the same as the one used in **B**. Angle, angular velocity, and acceleration are named as follows.

$$\alpha, \quad \dot{\alpha}, \quad \ddot{\alpha}$$

In this case, ℵ "Heth" is the $\angle CAG$ angle and is constant.

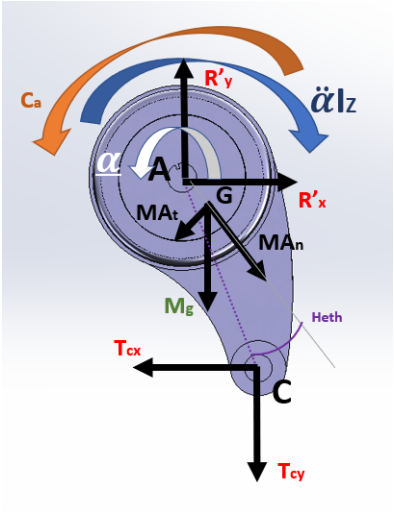

Figure 2.22: DCL Body 4

$$R'_X - T_{CX} - M_4 \cdot \ddot{\alpha} \cdot \bar{AG}_4 \cdot cos(\aleph) + M_4 \cdot \dot{\alpha}^2 \cdot \bar{AG}_4 \cdot sin(\aleph) = 0 \tag{2.14}$$

$$R'_Y - T_{CY} - M_4 \cdot \ddot{\alpha} \cdot \bar{AG}_4 \cdot sin(\aleph) - M_4 \cdot \dot{\alpha}^2 \cdot \bar{AG}_4 \cdot cos(\aleph) - M_4 \cdot g = 0 \tag{2.15}$$

$$\begin{aligned} C_a - T_{CX} \cdot \bar{AC} \cdot cos(\alpha - 270) - T_{CY} \cdot \bar{AC} \cdot sin(\alpha - 270) - M_4 \cdot g \cdot \bar{\Delta X} \\ - M_4 \cdot \ddot{\alpha} \cdot \bar{AG}_4{}^2 - I_{Z_4} \cdot \ddot{\alpha} = 0 \end{aligned} \tag{2.16}$$

2.5.5 Body 5

It is the last standard body for every TLP. At its base, there is a mechanism where it is possible to attach the different end effectors used to manipulate packaging, boxes, and products. Angle, angular velocity, and acceleration are named as follows.

$$\Xi, \quad \dot{\Xi}, \quad \ddot{\Xi}$$

In this case ⊐ "Beth" is the $\angle EGG_5$ angle and is constant.

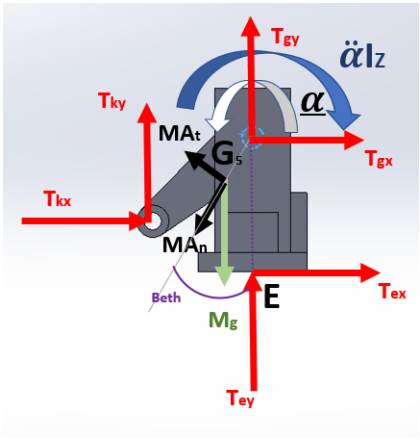

Figure 2.23: DCL Body 5

$$T_{KX} + T_{GX} + T_{EX} - M_5 \cdot \ddot{\Xi} \cdot G\bar{G}_5 \cdot cos(⊐) - M_5 \cdot \dot{\Xi}^2 \cdot G\bar{G}_5 \cdot sin(⊐) = 0 \quad (2.17)$$

$$T_{KY} + T_{GY} + T_{EY} + M_5 \cdot \ddot{\Xi} \cdot G\bar{G}_5 \cdot sin(⊐) - M_5 \cdot \dot{\Xi}^2 \cdot G\bar{G}_5 \cdot cos(⊐) - M_5 \cdot g = 0 \quad (2.18)$$

$$T_{EX} \cdot E\bar{G} - T_{KY} \cdot K\bar{Y}_x + T_{KX} \cdot K\bar{Y}_y + M_5 \cdot g\bar{\Delta}x - M_5 \cdot \ddot{\Xi} \cdot G\bar{G}_5{}^2 - I_{Z_5} \cdot \ddot{\Xi} = 0 \quad (2.19)$$

2.5.6 Body 6

It forms the last part of the second parallelogram; its primary goal is to maintain Body 5 in a horizontal position. $\lambda, \quad \dot{\lambda}, \quad \ddot{\lambda}$

The angle ℵ "Alaph" is defined in this case as $\aleph = \angle KWG - (\lambda - 270)$

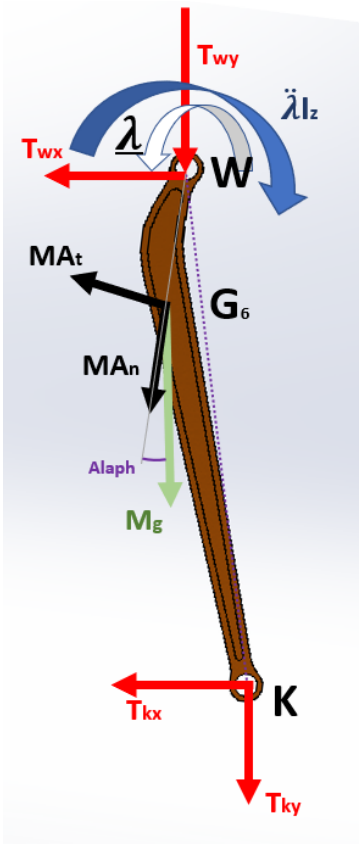

Figure 2.24: DCL Body 6

$$-T_{WX} - T_{KX} - M_6 \cdot \ddot{\lambda} \cdot W\bar{G}_6 \cdot cos(\aleph) - M_6 \cdot \dot{\lambda}^2 \cdot W\bar{G}_6 \cdot sin(\aleph) = 0 \tag{2.20}$$

$$-T_{WY} - T_{KY} + M_6 \cdot \ddot{\lambda} \cdot W\bar{G}_6 \cdot sin(\aleph) - M_6 \cdot \dot{\lambda}^2 \cdot W\bar{G}_6 \cdot cos(\aleph) - M_6 \cdot g = 0 \tag{2.21}$$

$$-T_{KX} \cdot W\bar{K} \cdot cos(\lambda - 270) - T_{KY} \cdot W\bar{K} \cdot sin(\lambda - 270) + M_6 \cdot g \cdot W\bar{G}_6 \cdot sin(\aleph) - M_6 \cdot \ddot{\lambda} \cdot W\bar{G_6}^2 - I_{Z_6} \cdot \ddot{\lambda} = 0 \tag{2.22}$$

2.5.7 Body 7

It is attached to the **D** shaft and is used to control the horizontal position of body 5. This body has a fixed orientation that does not change during motion. All the equations considering the velocity and acceleration are studied, but they are not reported in the Matlab solver.

$$\alpha_7, \quad \dot{\alpha}_7, \quad \ddot{\alpha}_7$$

The angle "He" (ℵ) is the angle used to define the center of mass.

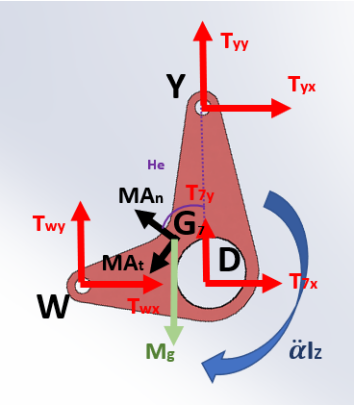

Figure 2.25: DCL Body 7

$$T_{WX} + T_{7X} + T_{YX} + M_7 \cdot \ddot{\alpha}_7 \cdot D\bar{G}_7 \cdot cos(\aleph) - M_7 \cdot \dot{\alpha_7}^2 \cdot D\bar{G}_7 \cdot sin(\aleph) = 0 \quad (2.23)$$

$$T_{WY} + T_{7Y} + T_{YY} + M_7 \cdot \ddot{\alpha}_7 \cdot D\bar{G}_7 \cdot sin(\aleph) - M_7 \cdot \dot{\alpha_7}^2 \cdot D\bar{G}_7 \cdot cos(\aleph) - M_7 \cdot g = 0 \quad (2.24)$$

$$-T_{WY} \cdot \bar{WD} - T_{YX} \cdot \bar{DY} + M_7 \cdot g \cdot \bar{\Delta x} - M_7 \cdot \ddot{\alpha}_7 \cdot D\bar{G_7}^2 - I_{Z_7} \cdot \ddot{\alpha}_7 = 0 \quad (2.25)$$

2.5.8 Body 8

Conclude the third parallelogram and have the hinge **X** that is not actuated.

$$\mu, \quad \dot{\mu}, \quad \ddot{\mu}$$

In this case, "Aleph" is defined as $\aleph = \angle YXG + (180 - \mu)$

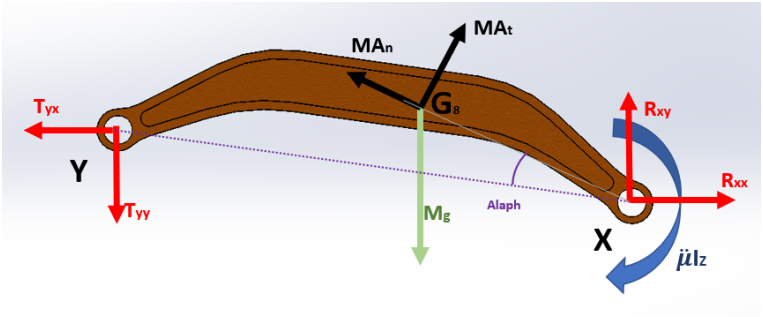

Figure 2.26: DCL Body 8

$$R_{XX} - T_{YX} + M_8 \cdot \ddot{\mu} \cdot X\bar{G}_8 \cdot sin(\aleph) - M_8 \cdot \dot{\mu}^2 \cdot X\bar{G}_8 \cdot cos(\aleph) = 0 \tag{2.26}$$

$$R_{XY} - T_{YY} + M_8 \cdot \ddot{\mu} \cdot X\bar{G}_8 \cdot cos(\aleph) + M_8 \cdot \dot{\mu}^2 \cdot X\bar{G}_8 \cdot sin(\aleph) - M_8 \cdot g = 0 \tag{2.27}$$

$$T_{YY} \cdot X\bar{Y} \cdot cos(180 - \mu) + T_{YX} \cdot X\bar{Y} \cdot sin(180 - \mu) + M_8 \cdot g \cdot X\bar{G}_8 \cdot cos(\aleph) - M_8 \cdot \ddot{\mu} \cdot X\bar{G}_8{}^2 - I_{Z_8} \cdot \ddot{\mu} = 0 \tag{2.28}$$

2.5.9 Body 9 - End Effector

It is the end effector, the nipper, that is interchangeable. It is modeled as an inverted cone to simplify the model. This method is enough to modify the height and its mass propriety to simulate the different end effectors.

$$\alpha, \quad \dot{\alpha}, \quad \ddot{\alpha}, \quad \ddot{X}_e$$

The DCL is reported in the following figure.

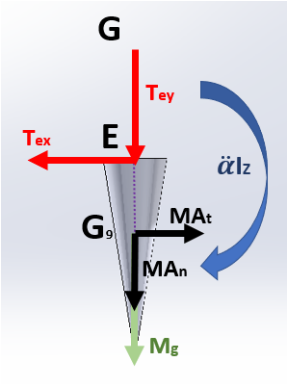

Figure 2.27: DCL Body 9

$$-T_{EX} + M_9 \cdot \ddot{X}_e \cdot G\bar{G}_9 = 0 \tag{2.29}$$

$$-T_{EY} - M_9 \cdot \dot{\alpha}^2 \cdot G\bar{G}_9 - M_9 \cdot g = 0 \tag{2.30}$$

$$-T_{EX} \cdot E\bar{G} + M_9 \cdot \ddot{X}_e \cdot G\bar{G}_9{}^2 - I_{Z_9} \cdot \ddot{X}_e = 0 \tag{2.31}$$

2.5.10 Reducer and Motor

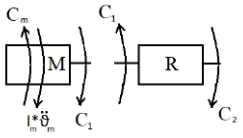

Figure 2.28: DCL Motor and Reducer

The relationship between the input torque in the gearbox and the output torque is as follows.

$$C_2 = C_1 \cdot \mu \cdot i \tag{2.32}$$

I have a reducer with $i = 83$ with an efficiency of $\mu = 0.95$. The equilibrium at the motor must also be considered. It is provided by the equation below.

$$C_m = C_1 - I_m \cdot \ddot{\theta_m} \tag{2.33}$$

2.6 End Effectors

Three different end effectors used in a secondary packaging machine are reported. The first one is used to take the experimental data utilized to validate the model.

The machine was chosen because it was in production and testing during the internship. It was also one of a batch of similar machines assembled by SACMI. Its performance, efficiency, and problems are known, and it works with standard TLP movements.

All the end effectors are simplified into an inverted cone. It is done to simplify the motion model and perfectly match the top of the cone to the path line. It is also fast the editing of its dimensions; this leads to using the same assembly with fast editing and not the change of the end effector.

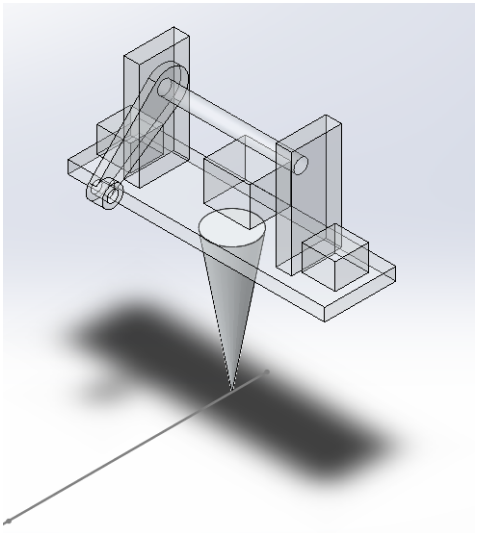

Figure 2.29: EE Cone

2.6.1 Lidding

It is the first TLP used in the cycle, named 1C1. It is responsible for taking the cover from the autonomous warehouse and bringing it to the boxing TLP. During movement, the carton that will be formed as a lid is sprinkled with glue. This part is useful for our analysis because it is a straight line motion with defined velocity.

The lidding end effector is shown in the figure below. It consists of four pneumatically operated suckers.

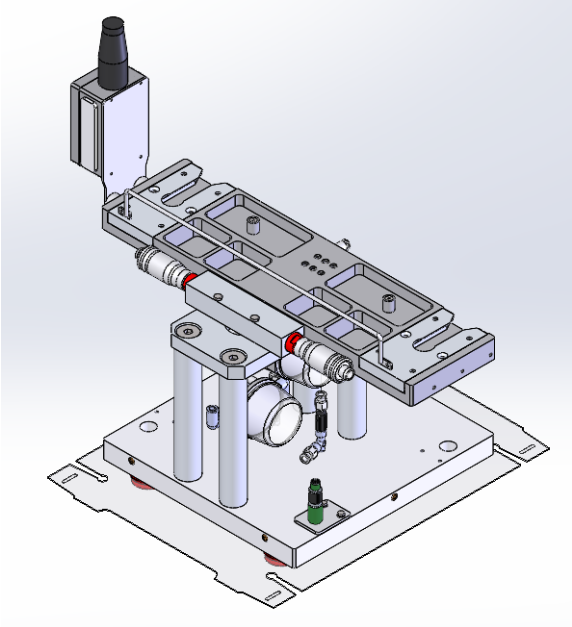

Figure 2.30: Lidding Model

The mass properties are shown in the table below. The centers of mass are referred to as points G, which allows simplification of the dynamic.

Mass	16.9	**Kg**
Center of mass	300	**mm**
Moment of inertia	509 067	$Kg \cdot mm^2$

2.6.2 Forming

The forming end effector takes the cover flat from where the first end effector leaves it and creates the box. The formation procedure is helped by different guides that help to close the box in a fixed sequence. The sequence of guides closes the box around the end effector, and the glue locks it in the position. The formed box is then taken by a shuttle mechanism that transfers it to the loading area. All displacements are made by the suckers that carry the boxes.

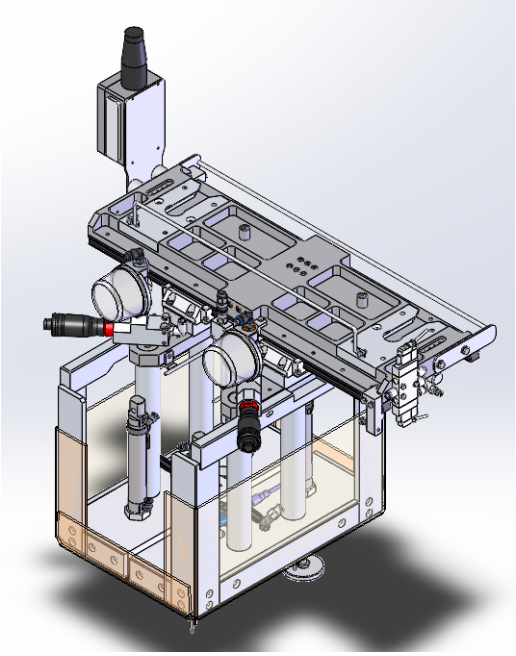

Figure 2.31: Forming Model

Mass	26.6	**Kg**
Center of mass	173	**mm**
Moment of inertia	826 370	$Kg \cdot mm^2$

2.6.3 Boxing

It is the last end effector that is considered. He has the scope to move the boxes from a conveyor to the cover formed before. The suckers are not present, and the boxes are moved only by two alloy plates.

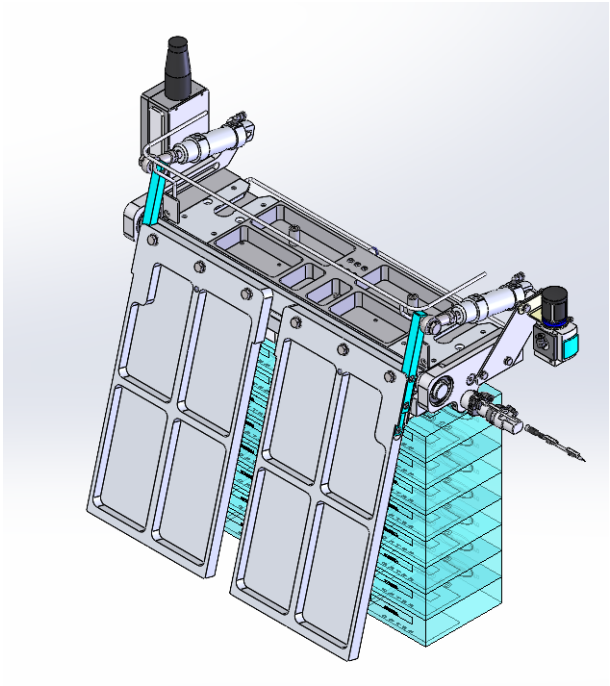

Figure 2.32: Boxing Model

Mass	20.6	**Kg**
Center of mass	150	**mm**
Moment of inertia	384 736	$Kg \cdot mm^2$

Chapter 3

Modelling TLP

3.1 Objective

The study's main objective is to create a model for the TLP. It will simulate the dynamics considering the trajectory planning given by the path and velocity. The model's results will be the torque and angular speed on the shaft of the motors. The data obtained will be correlated to the characteristic curve of the motor. This comparison will help the user understand whether the desired trajectory is reachable.

As said in the introduction, the TLP is a well-known manipulator, with its practical performance tested over the years. SACMI knows it can work well in defined areas and a given time. Known that the operation of the system engineering office is to prototype the machine. They know that if the cycle time is above the minimal cycle, it is possible to achieve this. These are the conditions under which the TLP works in more than 85% of the cases. The aim is to create a fast and reliable model that can be used to understand whether the TLP can reach its goal in the last 15% of the case.

The office needs a fast and user-friendly model created in the available software. It must have input data related to the end effector, the trajectory, and the cycle time. It will output the torque needed to achieve the input trajectory. The software used in the office are mainly two: Solidworks and Excel. They do not have the possibility of using Matlab, so the university license is used only to validate the model. Solving a non-linear differential dynamics system with more than fifteen trigonometric equations in Excel is not feasible.

The study was carried out using a Solidworks Motion model and a Matlab model and then comparing the data obtained with actual experimental data.

3.2 Experimental Data

Experimental data are collected and evaluated from two TLPs that move on a defined trajectory. Up to 24 different data are collected from the two TLPs.

The PLC that controls the TLP reads and extracts only one data at a time. So all the data can be read every microsecond. The extracted file was a *.trace* file with comma-separated values. The data are from continuous cycles taken in 10 minutes of operation. They must be clean from the non-useful data and extract the motion in a single cycle.

The cycles are similar but can differ by milliseconds due to waiting time. An average cycle is used as a standard to evaluate the two models.

The data collected are the following.

Time instant	[**microSecond**]
Horizontal position	[**mm**]
Horizontal velocity	[**mm/s**]
Motor A position	[**Grad**]
Motor A track	[**Grad**]
Motor A current	[**milliAmps**]
Motor A angular velocity	[**Grad/s**]
Vertical position	[**mm**]
Vertical velocity	[**mm/s**]
Motor B position	[**Grad**]
Motor B track	[**Grad**]
Motor B current	[**milliAmps**]
Motor B angular velocity	[**Grad/s**]

The trajectory's main characteristics are described in the data. That information will be used to feed the models. The program reported in *Appendix C* is used to extract the data and create a unique matrix that Matlab and Solidworks can read. The main characteristic of the cycle will be reported in the following subsection.

3.2.1 Trajectory

The trajectory followed by the TLP is plotted in the Matlab figure below.

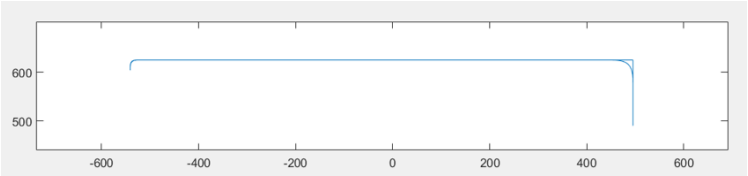

Figure 3.1: Trajectory

As is possible to see, the trajectory is straightforward, with one horizontal and two vertical movements. The cycle time is less than 3 seconds. Measurements are reported in millimeters.

The first evaluation that is possible to do is that the orientation system is different from ours. The X-axis's origin is fixed in motor B's shaft, while the Y-axis is from the ground to the motor with some offset. This offset change for every TLP.

In our test case, this offset concerning our is **327 mm**. An Excel program is used to simplify the creation of the Solidworks and Matlab models. As input the main marker points of the curve, it gives the path plot as output. If it is inside the blue area, its kinematic is reachable. In particular, our horizontal axis is set at 327 mm concerning the X-axis of the working system.

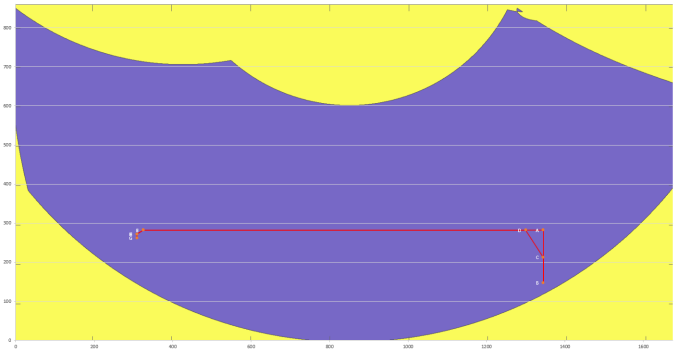

Figure 3.2: Trajectory in our reference system

The horizontal trajectory is used to validate our model. It is used to avoid possible problems in Solidworks during forward-backward movement on the same spline.

3.2.2 Velocities

The velocities on the X and Y axes are plotted below and are measured in **mm/sec**. The trapezoidal velocity strokes help create a Solidworks model that the program can easily differentiate. In Solidworks is present a predefined function that represents the trapezoidal velocity.

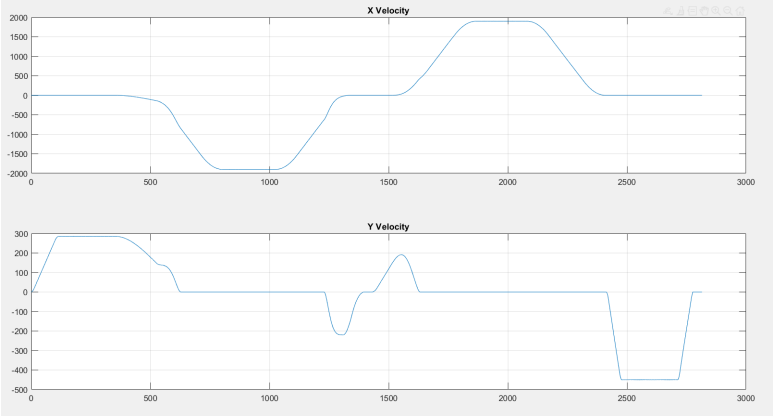

Figure 3.3: Velocity XY

In this case, is present a nominal cycle with a passage under a glue sprinkler. This passage occurs when there is zero vertical velocity and constant horizontal velocity. The glue will be used to close the box.

The position and velocity data of both motors are collected. These are useful for validating the kinematic model in Matlab and Solidworks. The data from both motors are plotted here.

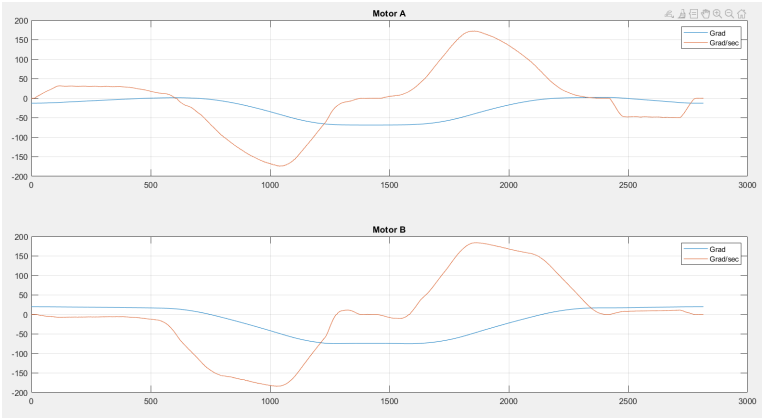

Figure 3.4: Motor Position and Velocity

3.2.3 Torque

The collected data do not include the torque, but the technicians gave us technical details. The torque can be correlated with the equivalent current using the parameter **Torque Constant** K_t. The technical data of the motors give this parameter. (Figure 2.14)s

The kinematic chain that connects the two motors must be considered. It can create some force and torque relation. Due to the Solidworks limitation, those relations are not considered.

The alternate current oscillates around a medium value, and the mean measure value, obtained from the *movmean* Matlab function, is reported in the graph. It gives the moving mean value for a vector as output, and the positive integer scalar K computes a centered moving average by sliding a window of length K along the vector.

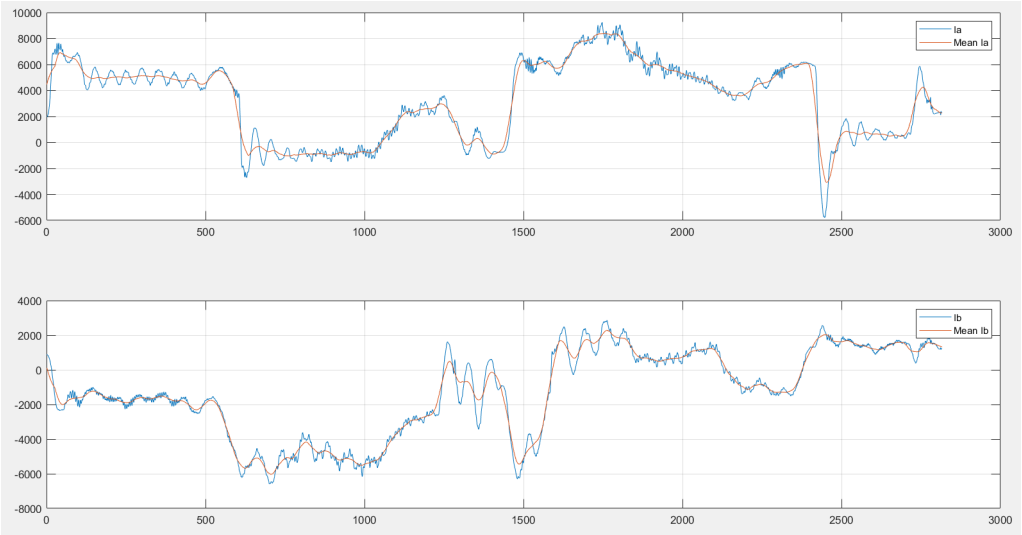

Figure 3.5: Motor Current

3.3 Solidwork Model

The model in Solidworks is simulated with Motion. Some limitations in the analysis are reached, but it is possible to appreciate similar trends regarding the experimental data. For the first analysis, all the mass proprieties are overwritten. We found that this is not the best working condition.

This section reports the steps to achieve a complete Solidworks analysis. The goal is to start from the input path and trajectory to have the torque of the two motors as an output.

3.3.1 Assembly the model

First of all, the assembly is simplified as much as possible. The simplification means removing all the shaft, screw, and supply from the assembly. It is done to make the manipulator easier to simulate and move.

Every bodies must have the same significant dimension and shaft position. To simplify the assembly the non-useful parts are delated. The origin of each body is located on the first shaft. It is done to assign the center of mass to the body quickly. In total, nine main components and the end effector are present.

The assembly is made from a blank model. The first body added is the main base or *Body 9*, which is the only body that does not move. Its origin is mated to the assembly origin and FIX its position. The fix command blocks all movements and will allow free movement of the TLP.

All other mates are **HINGE MATES**. As explained in the first chapter, those mates have the same effect as a concentric mate plus a coincident mate. In some cases, only concentric mates are used to avoid redundancy. In the further motion simulation, redundancy must be taken into account.

After the assembly of the model, it is time to set up **Motion Analysis**. The passages needed to simulate the motion are fixed and must be followed in a precisely ordered way. The passages to do are explained in order; if a passage is forgotten or changed, it is possible to have errors in the solution of the simulation.

3.3.2 Pre-simulation

Before the simulation, the environment is set up. To avoid deleting or corrupting the assembly, a pack and go **MAIN** assembly is created. It will be copied and pasted for every simulation. In this assembly, the minimum number of mates are set and all initial conditions are deleted. It is a blank page from which the model can be simulated without error.

The only file to open from the folder is the assembly named TLP MAIN Motion. All other files will be opened from the Solidworks feature manager design tree. The first item to modify is the spline that will be followed during the motion. The *Open Part* option must be used on the Path of the assembly. It will open a new Solidworks page where it is possible to modify the path.

The Fit Spline function must be used to create the path: *Tools - Spline Tools - Fit Spline*. However, the geometry where the spline will fit must be created first.

Geometry creation must be done according to the desired path. The design of the entire path can be done in the excel file. Then the path segment I want to simulate can be reported in Solidworks. It must be considered that it is possible to simulate only one-way movements. It is not possible to simulate a cycle on the same path. That must be taken as advice for every simulation. So to simulate a complete cycle, the gone and return analysis must be simulated in two different cycles.

The path can be modified with "Edit the Sketch" of the *Modify* component, as shown in the figure below.

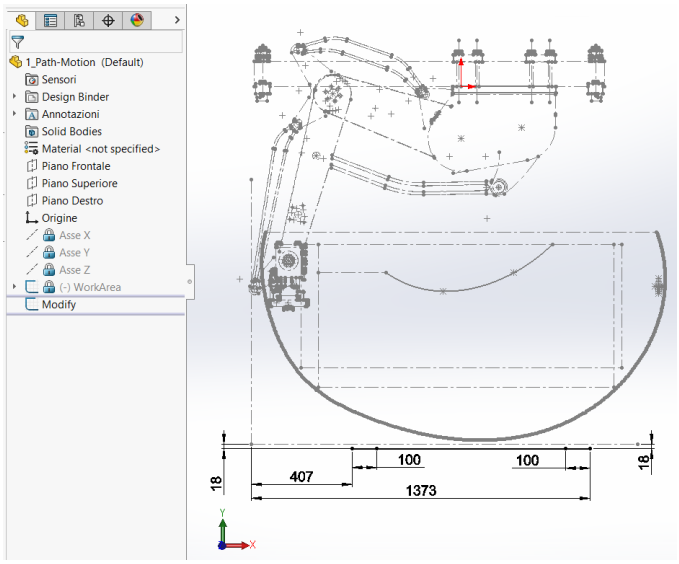

Figure 3.6: Path Modification

As can be noticed, the spline, in this case, is a design of two segments of 100mm in length. The segments positions are defined from the assigned path to follow, plus the length of the end effector. This modification is done directly in the Excel file from which I can take the distance.

As shown in Figure 3.6 the path is located outside the work area because the work area is not defined at the end effector but at the base of body 5. If a linear spline is needed, it is possible to modify the dimensions of the present one directly. However, if a more complex path is needed, it must create from a new sketch and design from the beginning. Next, is explain how to design the path from zero:

- Delete the old sketch.
- Create a new blank sketch.
- Design the new "bone" of the path as seen in the following figure; the path represents pick-up, move-up, and place movements with defined dimensions.

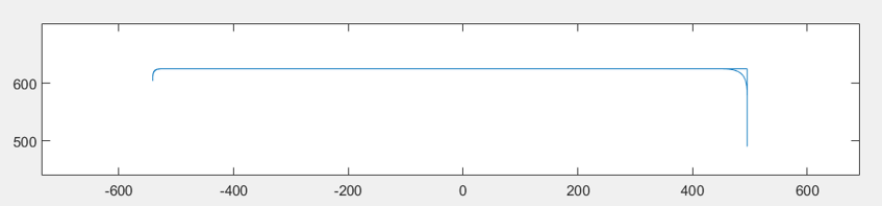

Figure 3.7: Spline Bone

- The path must be **OPEN** and not connected: it must have an open passage in a straight-line position. This passage will be close by the Fit Spline function.
- Define all the distance; this is useful for later modification of the position and dimension of the path without creating a new one. Because when the spline is fitted, it will be impossible to add distance. It is only possible to modify the one set before the fitting.
- Select all the segments and Fit the Spline.

During the creation of the spline, there is a property window where it is possible to modify the characteristic of the spline. The "Closed spline" parameter must be unchecked to have a one-way path. Also, the tolerance must be increased up to 0.15 **mm**.

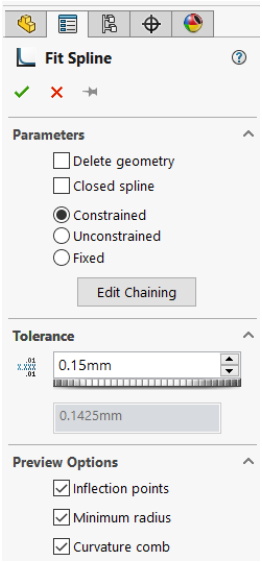

Figure 3.8: Fit Spline Propriety

If the path is created correctly, save, close and return to the assembly. Here the path mate must be updated.

In the Mate folder, the one with the clips, the *PathMateModify* must be selected and the edit feature menu open. If the initial spline is modified, it is possible only to check and see that the mates are correct. Instead, if the spline is created from a sketch, it must be mated at the end effector's vertex. The Path selection highlighted in the following figure must be modified with the new spline.

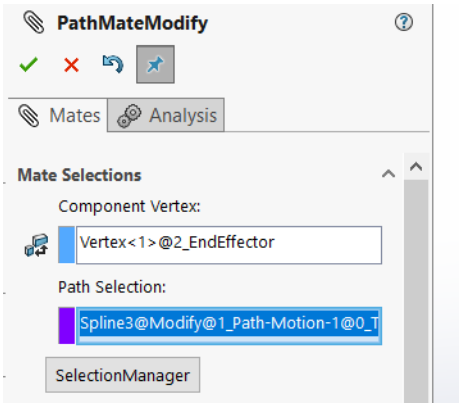

Figure 3.9: Path Mates

By adding the spline to the manipulator, it must be controlled whether the dimensions of the end effector are correct for the analysis. It can be done by opening the End Effector body and editing the feature of the *CONE* revolution. Here, it is possible to modify the cone length from the sketch. As is possible to see in the figure below, the value of 185.5 mm is the distance from the center of part 5: it must not be modified. In the figure below, the only dimension editable is the length of the cone: **300 mm**.

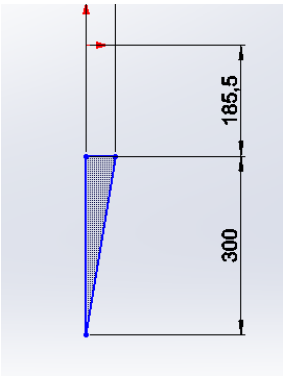

Figure 3.10: End Effector

Before closing the end-effector body, the mass property must be modified: *Evaluate - Mass Properties - Override Mass Properties*. The mass and center of mass can be taken from the Solidworks assembly of the end effector. The moments of inertia are not modified due to calculation problems for the motion.

Saved and closed all windows I will return in the assembly and *rebuild* it (*Ctrl + B*). To clean the workspace of the assembly, the Work Area can be hidden as all other sketches that are not needed. Now the user can move the end effector along the path the manipulator will move with it.

Now the initial configuration condition must be set for the TLP. This configuration is achieved by setting the angle of the motors at the starting point.

By default, *Angle A* and *Angle B* are defined. They are suppressed and must be deleted and assigned again. They are left in the assembly as an example of the passage to do. First, the TLP must be positioned at the beginning of the path, in the start position, then the angle mates can be assigned.

The angle of mate A connects the right plane of the assembly to the top plane of part 4. They are selected from the design tree of the assembly and as the angle mates. The value of the angle mates is automatically assigned and does not need to be modified.

The mate B angle connects the assembly's right plane and the top plane of part 1. Again the dimension of the angle is automatically assigned.

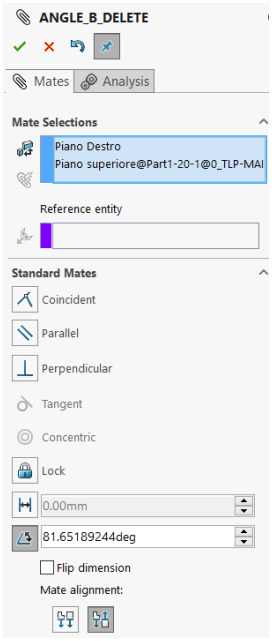

Figure 3.11: Initial Condition Angle Mates

Now I have to rebuild the assembly and prepare the environment for motion analysis. During the building, I can have warning errors because the assembly is over-defined. I will not consider this error and close it.

After rebuilding the assembly, it is possible to suppress the angle mates and move to the kinematic study. Pay attention that all those modifications must be done in the Model window.

Figure 3.12: Windows

3.3.3 Inverse kinematic

Moving to the Inverse window is possible to assign at the end effector a path mate motor and study the motion. In particular, the motion at the motors' shaft is obtained. It will be used in the direct kinematic.

First of all, it must be verified that we are in the *Motion Analysis* window, here is possible to assign the gravity conditions and the motors. In the available inverse kinematics, gravity is already defined. If a new one is created, it must be defined again. To assign gravity, there is a command called *Gravity Parameter*, and the user must pay attention to the direction of gravity.

Using the predefined kinematic can be easy because the Results and Plots are defined, and most redundancies are deleted. However, before setting up the kinematics, it is better to delete the motion result to clean all the data.

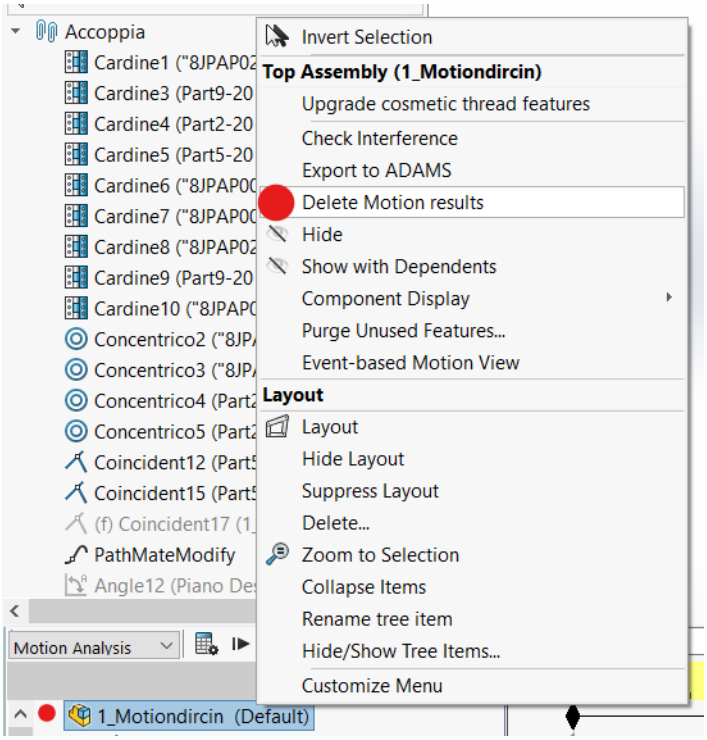

Figure 3.13: Delete motion Result

Now the path motor must be defined. To select the motors, the steps to follow were explained in chapter **1.4.6** for the path mate motor. The easiest way to assign the displacement that the motor must observe is to use the function builder and the segments' propriety. In this way, Solidworks will directly assign an internally defined mathematical function that can be easily differentiated during the motion.

These functions can be defined by imposing the initial and final time, the value needed to reach the end time, and the segment type. The interface is like the following table.

Start X	**End X**	**Value**	**Segment Type**
-	-	0	Initial
0s	xx sec	xx mm	*Function*

Table 3.1: Input Segments

The mathematical functions available to interpolate the data are as follows.

- Cubic
- Quarter-Sine
- Quarter-Cosine
- Half-Cosine
- 3-4-5-Polynomial
- 4-5-6-7-Polynomial
- 5-6-7-8-9-Polynomial
- Cycloidal
- Quadratic
- Linear

The main disadvantage of using a segment is that if more rows were defined, the velocity between one row and the following would return to zero. This zero velocity point imports into the acceleration and Jerk graph discontinuities that will generate errors in the direct kinematic. The best way to create a motion is to use a single segment.
The Data Points builder can be used if the trajectory is impossible to simulate in a single segment. With its use is possible to define the time at which the end effector must arrive at a defined spline length. The spline is defined incrementally from 0 to the last point.

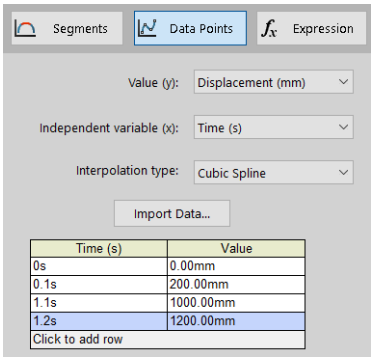

Figure 3.14: Data point builder

The actual spline's length differs from the original segment's sum. That is due to the fillet that can be present at the angles of the trajectory. The difference is not so significant, but to avoid not covering the entire path, it is possible to convert in Excel the actual length with the defined data points. The accurate measure of the spline is defined through the *Evaluate - Measure*.

A tip is always to define the position and not the velocity of the motion. As is possible to see in figure 3.17 some mates are deleted to reduce the redundancies as much as possible, but they are present in any case.

Now that all the motors are defined, it is time to simulate the motion. The duration of the simulation can be modified in the Edit Time, it can be set at a higher time, but it is better to set it at the end time of the motion.

First, I have to position **timeline** at time 0. Then both the angle defined before must be **Unsuppress**, the assembly is **rebuild**, errors can occour. Then both angle mates are **Suppress** and the motion study is **calculate**. The simulation is correct if the trace path plot follows the spline.

With a simulated study, it is time to extract the shaft position data to feed the motors in the direct kinematic. The hinges or the shaft of the motors are selected to extract them. The displacement can be easily imported in the following motion if the shaft is used. Instead, if the hinges are used, the direction of rotation in the direct kinematic must be controlled. In general, hinge data are more precise.

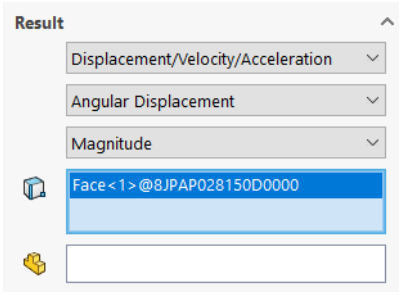

Figure 3.15: Plot Options

If I prefer to extract the result from the shaft; the conditions above are those I have to set in the result option. The face in this case is the shaft of the motor B. The same must be done for motor A.

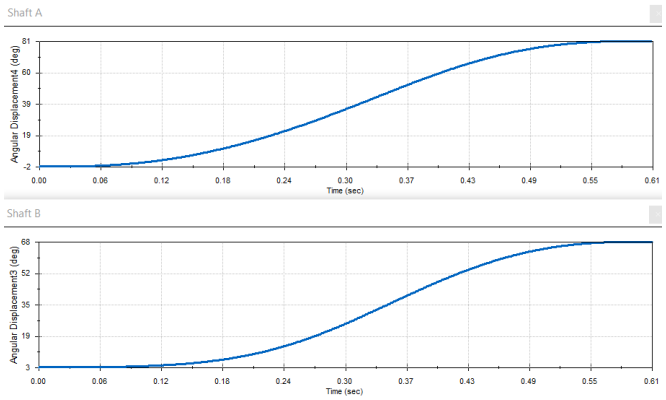

Figure 3.16: Shaft Displacement

From the graph, it is possible to extract the position data as a CSV file. Please right-click on the graph and save it in the same assembly folder. They must be saved precisely. If the data are confused in the next part can create problems in the simulation.

The files are saved in CSV format. The title occupies the first two rows. If the files are updated directly in Solidworks, there are no problems. However, the first two rows must be deleted if they are analyzed in Matlab.

3.3.4 Direct kinematic

In the direct kinematic study, the motor's displacement will be inputted, and the torque needed to follow the desired trajectory will be obtained as the output. Again there is a predefined motion study, but it can be helpful to create a new one due to possible errors. The mates are simplified to avoid redundancy. In this case, more mates are deleted. There are no redundancies.

The mates are reported in the figure, and it is possible to see which must be deleted in the different studies.

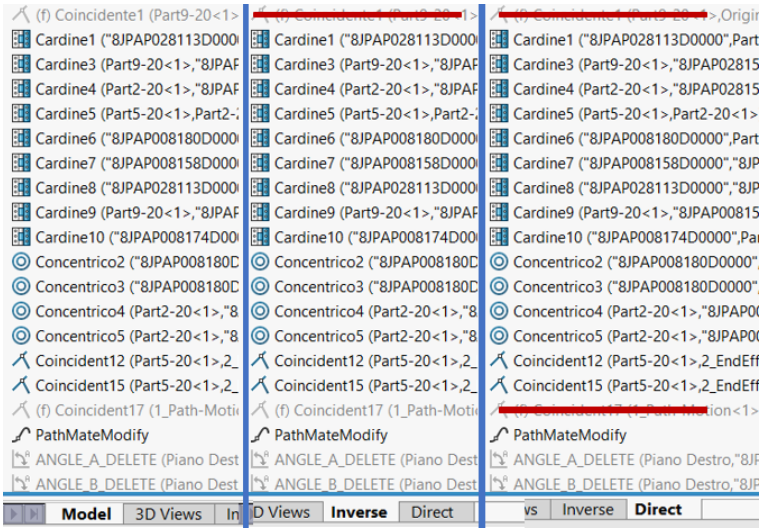

Figure 3.17: Mates to Delete

As before, if a new motion study is created, the gravity must be added, and the Path Mate must be suppressed to leave the end effector free to move.

It is time to assign the motors to the shaft and upload the data points saved from the inverse study. Data are imported as displacements, and the orientation of the motor must be controlled. Usually, the first one automatically assigned is correct, but a fast check is reasonable. If the displacements are extracted from the shaft, they can be directly assigned to the motor shaft without changing the orientation of the motion. Instead, if they are extracted from the hinge, the orientation of the motor must be changed.

When CSV files are uploaded on the right side of the function builder, it is possible to see all the displacement, velocity, acceleration, and jerk graphs. With a tight control, it is possible to determine whether the passage performed in inverse kinematics gives the correct result or not.

The image below reports two sets of acceleration and jerk graphs.

In the second row, the graphs have noise. It is because velocity, acceleration, and jerk are obtained as derivatives of position. If the position is made from a non-mathematical function or has some noise, then it is possible to have errors, which will also be present in the torque.

The first row of the graph is obtained from a mathematical function directly defined by Solidworks. It is possible to see that it is smoothed and linear concerning the first one.

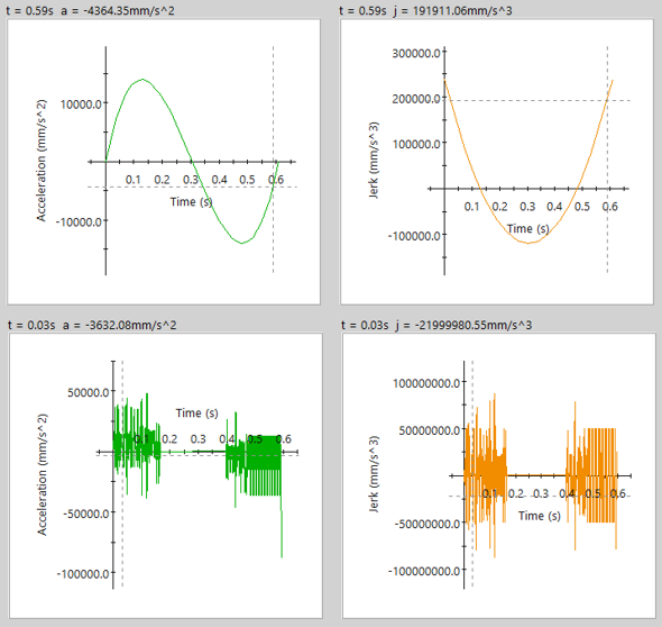

Figure 3.18: Disturbance

If I am in the case with noise, it is better to define another equation or clean the input data, removing the oscillating part. Data can be uniform with the Matlab function *movemean*.

The model can now be simulated by imposing the motion on both motors. The passages are the same as before. They must be followed in the same order.

First, I have to position **timeline** at time 0. Then I must **Unsuppress** both the angle defined before and **rebuild** the assembly, I can have errors. Then again **Suppress** both angle mates and **calculate** the motion study.

The simulation is correct if the trace path plot follows the spline. If not, there are probably some errors in the procedure or old data in the motion result.

In these cases, creating a new motion study from sketches is better. Another possibility is that some incorrect data are used for the motors, perhaps inverted in the direction of the motors.

The result can be accepted or the simulation repeated. The distance between the desired path and the real one can be estimated using the measure features. If the path is similar, the motion can be accepted and see if the torques of the motor have an acceptable trend or not.

Two new plots must be created to extract the torque from the motion. The objects to select are directly the motors on the Y component. The CSV file can be extracted from the graph and analyzed in Excel. The torque can have disturbance due to the error, as explained before.

3.4 Matlab Model

We create a Matlab model of the dynamic to validate the data obtained in the Motion analysis. It will be used to solve the same path and compare the results. The model starts from a trajectory with a defined position at a given time and gives the torque needed to follow the trajectory as output.

The program used to verify the value obtained in the actual case is reported in **Appendix D** . In particular, it is the program used in **Test2** to validate the models.

The software's different sections and the functions used are explained.

3.4.1 MAIN

The main is where the input conditions are created and elaborate. The model for this case is created for a linear trajectory; in this way, only the X position must be changed. In the reported model, the initial and final positions are externally defined and uploaded as a CSV file with position and velocity over time.

If the CSV file is unavailable, the trapezoidal trajectory can be created using the following commands.

```
EXi=407;        EXo=1373;
EY=282;

EndTime = 0.60

Wx=[EXi EXo];
Nsamp=EXo-EXi;

[Xe, dXe, ddXe, Te, PP] = trapveltraj(Wx, Nsamp, "EndTime", EndTime);
```

In this way, it is defined a trajectory followed by a trapezoidal velocity. The imposed inputs are the initial and final positions and the time the path must be followed. The Matlab function *trapveltraj* generates piecewise polynomials through multiple waypoints using trapezoidal velocity profiles. It computes a trajectory through a given set of input waypoints. The function outputs positions, velocities, and accelerations in the given time samples.

When the position and velocity vector are calculated, it is possible to determine the angle that defines the position and orientation of the TLP. From these angles, their velocity and acceleration can be derived.

The same function reported in the Appendix A is recalled to define the angle. It is fed with all the X and Y positions and outputs the four vectors of the angle

in radiant. The translation added is to start the angle from the horizontal position, not from the position defined in figure 2.9. It is helpful for the derivation of velocity and acceleration.

The derivation is carried out in a *for* cycles with the *diff* command. The position vectors are differentiated from the time vector defined at the beginning. The command *diff* gives as output a vector with a value less than the original. Initially, zero as the first value is inserted, but then it changes and inserts a value different from zero.

Next, the values at the angles $\theta \quad and \quad \mu$ are assigned. They are equal respectively to $\delta \quad and \quad \gamma$. With all those angles, velocity, and position defined, the dynamic can be simulated.

To simulate the model, two functions are created to have a cleaner and more comprehensive program: DINAMICA and DIN.

3.4.2 Functions

The first function to be called is "DINAMICA" in Appendix E. This function reported the main equation of the dynamics that are possible to resolve. The equations insert are the following:

	Eq
Tex = ...	2.29
Tey = ...	2.30
α = ...	2.31
Tcx = ...	2.11
Tcy = ...	2.12
Ca = ...	2.16
Cb = ...	2.07

Table 3.2: DINAMICA function

The other values are obtained by solving a system with all the other equations. This system is solved using the *fsolve* command and the "DIN" F function.

All the other functions in the form of $F(x) = 0$ are inserted in the DIN function. The decision to divide the system is made to lighten the computation. The equations not needed are not embedded in the system.

It is a system of 14 equations in 14 unknowns. Matlab can also solve it with a standard calculator in a reasonable time.

	Eq
F(1)	2.08
F(2)	2.09
F(3)	2.10
F(4)	2.17
F(5)	2.18
F(6)	2.19
F(7)	2.23
F(8)	2.24
F(9)	2.25
F(10)	2.13
F(11)	2.20
F(12)	2.21
F(13)	2.22
F(14)	2.28

Table 3.3: DIN function

Dimensions are defined in a shared global function called directly by those who need them. This function is used to modify the data only in a position and not in every function. It is also helpful to avoid errors or mistakes if the data must be corrected.

The MAIN model outputs are the two torques the motors must give. The torque can be extracted as a CSV file with the command *csvwrite* to analyze and compare their values during the tests.

Chapter 4

Test and verification

This chapter aims to illustrate the tests done to verify the models in Motion and Matlab and understand their limitations. We decide to create three different tests. The first is to verify the kinematic model and compare the angular displacements and the angular velocity of the models. The second is to verify from actual data that the models have a similar trend. The third is to simulate the Motion model on a new trajectory defined in Solidworks and specify proper procedures that must be used.

4.1 Test 1 - Kinematic

4.1.1 Objective

The objective of the first test is to verify that the kinematic qualities of the models are similar. The angular position and velocity of both motors are verified as output, giving as input the same path to follow under the same conditions. The results will be compared to validate the kinematics.

4.1.2 Procedure

The simulation is a straight path from A to B; the start and end values of the line are reported in the following table.

	X mm	**Y mm**
A	407	282
B	1373	282

Table 4.1: Test Path

For the path, actual data of angular position and acceleration of the motors are known as the end-effector velocity. The positions given by the real data are imposed to follow the path. The trajectory is affected by noise from errors in tracking the expected value. The noise is possible to see directly in the acceleration and jerk. As explained in the previous chapter, this noise will affect the data in the direct analysis. The graphs with noise are possible to appreciate in the following figure.

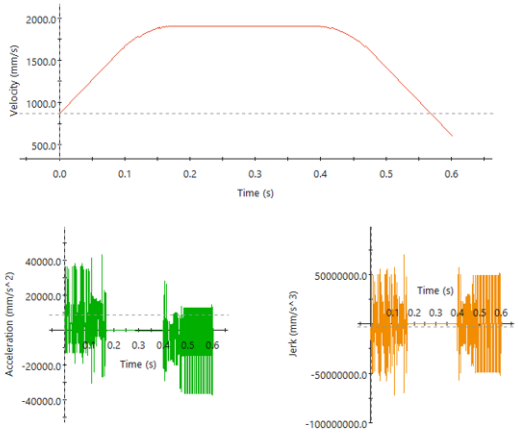

Figure 4.1: End Effector Velocity, Acceleration and Jerk

The displacement and acceleration graphs are obtained and shown below.

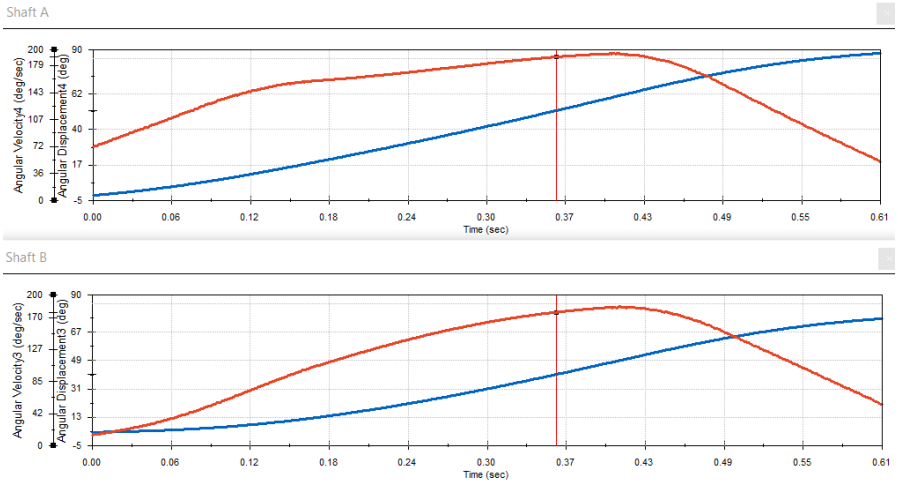

Figure 4.2: Solidworks shaft displacements and velocity

From the graph, the CVS files are extracted. They will be analyzed later in Matlab. For Matlab, the whole program is not executed, only the first part until the end of the "Velocity and acceleration angular profile" section. The value needed to compare the PVA matrix with the other data is extracted with *csvwrite*.

The following Matlab script extracts the data from the CSV file and uploads them to the MAIN program.

```
P = csvread('RealData.CSV');      %read CSV file

Te  = P(:,1)';                    %time
Xe  = P(:,2)'+EXi;                %position
dXe = P(:,3)';                    %velocity

ddXe = diff(dXe)./diff(Te);       %acceleration
ddXe =[ddXe(1) ddXe];
```

Exported all the data, a Matlab plotter is created to compare them. All data is uploaded and saved in different vector arrays. The displacement of motor A is initially plotted without adding an offset. The following is the first graph obtained.

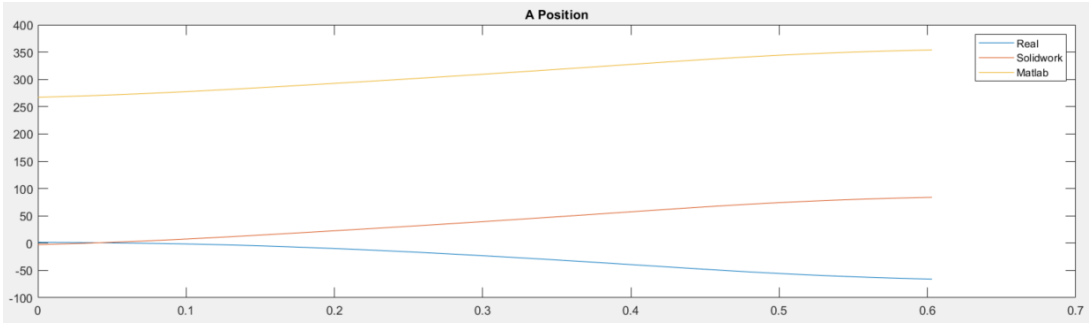

Figure 4.3: Motor A - Graph 1 - Displacement [Grad - Time]

As is possible to see, different offsets must be taken into account. It must decide which reference system and report all the other systems to it. The Solidworks model reference system is set as the reference one. All other models are scaled to it.

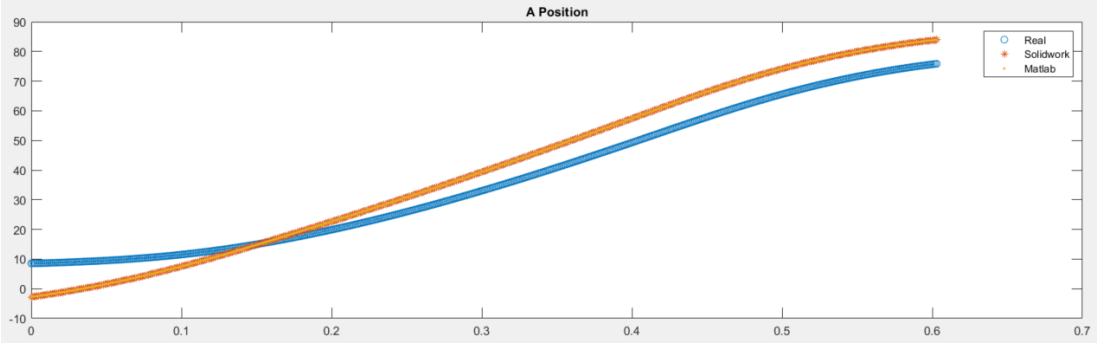

Figure 4.4: Motor A - Graph 2 - Displacement with offset

At first sight, it is possible to notice that the real line differs from the simulated one. The same error is found in motor B. By sharing the problem with the technical office, we ensured that the real data were exchanged. So all the data I assign to motor A are, in reality, of motor B and vice versa. The final graph for the displacement of motor A is as follows.

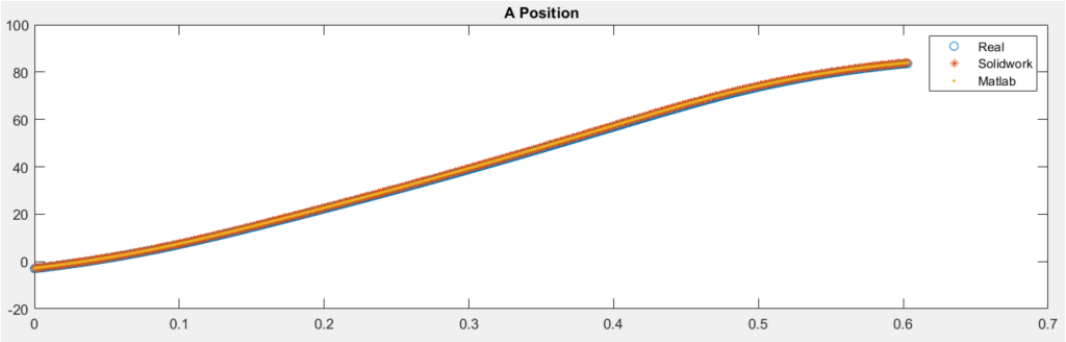

Figure 4.5: Motor A - Graph 3 - Displacement with offset

The offsets concerning the Solidworks reference system are reported in the following table. The same procedure is performed for motor B.

Motor A	OFFSET
Real	$260 - \alpha$
Solidworks	α
Matlab	$\alpha - 270$

Motor B	OFFSET
Real	$-\gamma - 10$
Solidworks	γ
Matlab	$\gamma - 180$

Table 4.2: Motors Offset

All the offsets are defined, and the motor assignment is changed for the real data. The entire graphs can be plotted.

The following graph reported the displacement and velocity of both motors.

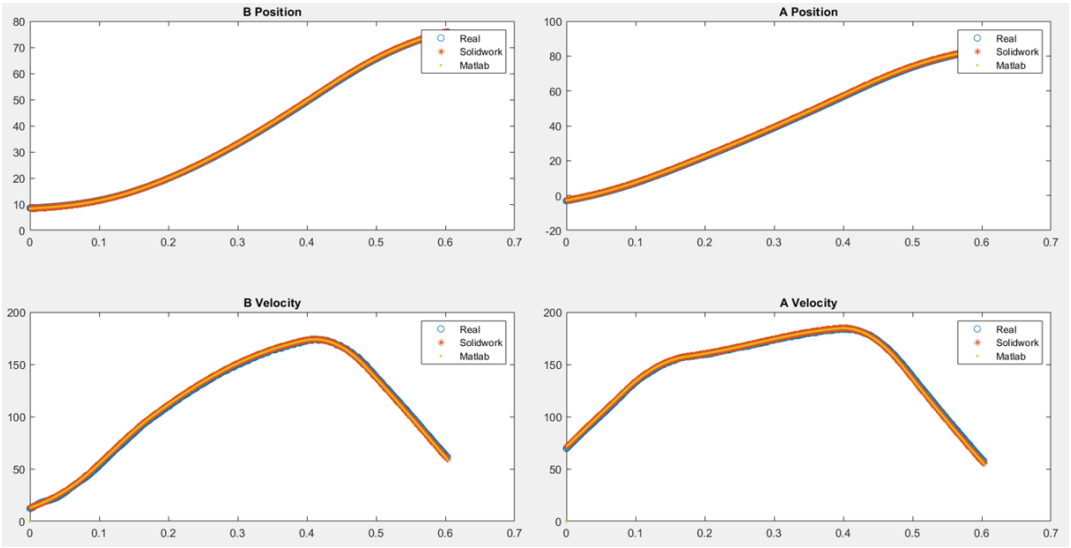

Figure 4.6: Motor A/B - Displacement and Velocity

4.1.3 Upshot

As seen in the figure 4.6, it is possible to say that the kinematic model corresponds to the real one. The two models follow the same trend in a defined path as in a real TLP.

This test also helps us understand how the motor is assigned to the collected data. The first test is passed without any problems and limitations.

4.2 Test 2 - Dynamics

4.2.1 Objective

This second test aims to verify that the dynamics of the models have a trend similar to the real one. It must verify that the torques obtained as the output are similar or follow a similar trend. The torque of both motors will be verified as output, given as input the same path of Test 1. The results are compared to validate the dynamic.

4.2.2 Procedure

The same passages done for Test 1 are used for the first part of the simulation. The displacement for the motors extracted as a CSV file will be uploaded as input in the data point for both motors. We notice immediately that the graphs of acceleration and jerk have noise. The noise will arrive directly at the torque that we want as the output.

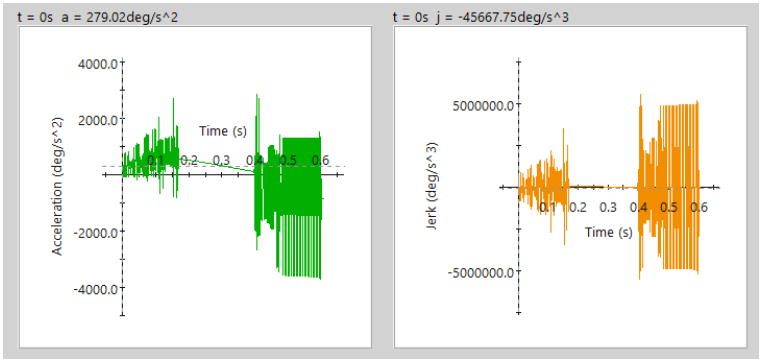

Figure 4.7: Motor B - Acceleration and Jerk with noise

Simulating the model, we immediately obtain a torque with this curve.

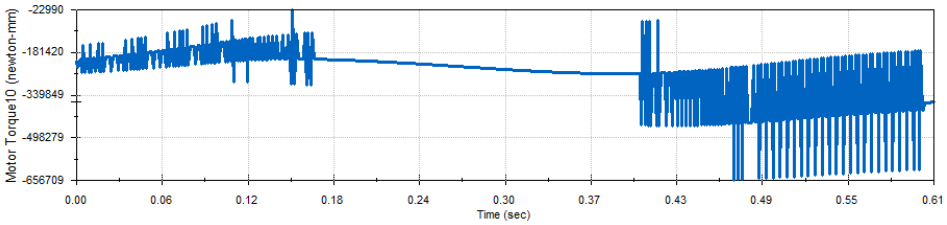

Figure 4.8: Motor B - Torque with noise

The torque obtained can be compared, but cleaning the input data and simulating the process from the beginning is better.
The *movmean* command is used to clean the data. It helps us create a more uniform curve easily differentiated by the Solidworks and Matlab models.

Using uniform data helps avoid discrepancies between the simulated path and the real one. The noise is also due to errors or lapses in the procedure used to simulate the motion. The steps are those reported in Chapter 3.3.

As is possible to see in the figure below, we have a discrepancy between the path followed by the end effector and the expected one.

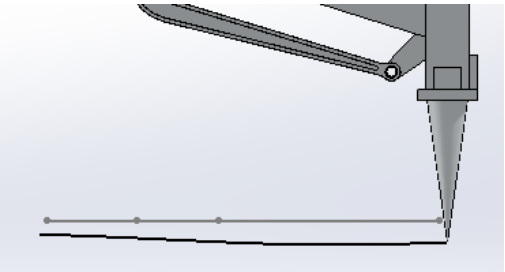

Figure 4.9: Different Path

After averaging the values, the steps are repeated, and a simulation with some noise is obtained, likely with less intensity.

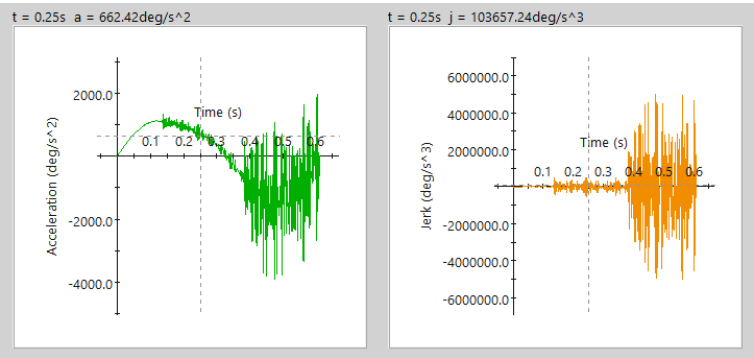

Figure 4.10: Motor B - Acceleration and Jerk with less noise

To avoid having the same torque error, the data are analyzed and cleaned again before importing them into Solidworks. Also, creating a simulation from a blank motion study helps avoid superposition errors.

Finally, the torques obtained are smooth and regular and can be scaled and faced in Matlab. Their performances are reported here.

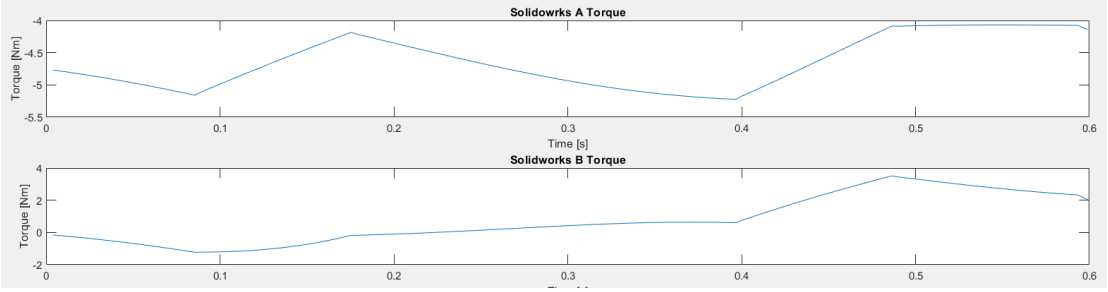

Figure 4.11: Motors Torque

As is possible to see, both torques are now smooth but have non-regular trends. The torques are collected in different positions and with different scales. The data can be from the motor shaft, the reducer shaft, or the input current. The unit of measure can be **Nm, Nmm or Arms**. All these differences must be taken into account when comparing trends.

The data as CSV files are imported to create the torque graphs in Matlab. They are analyzed considering the variation mentioned before. The graphs are made with two subplots and the superposition of different curves. In addition, the command *movmean* is used to visualize the trend of the real torque.

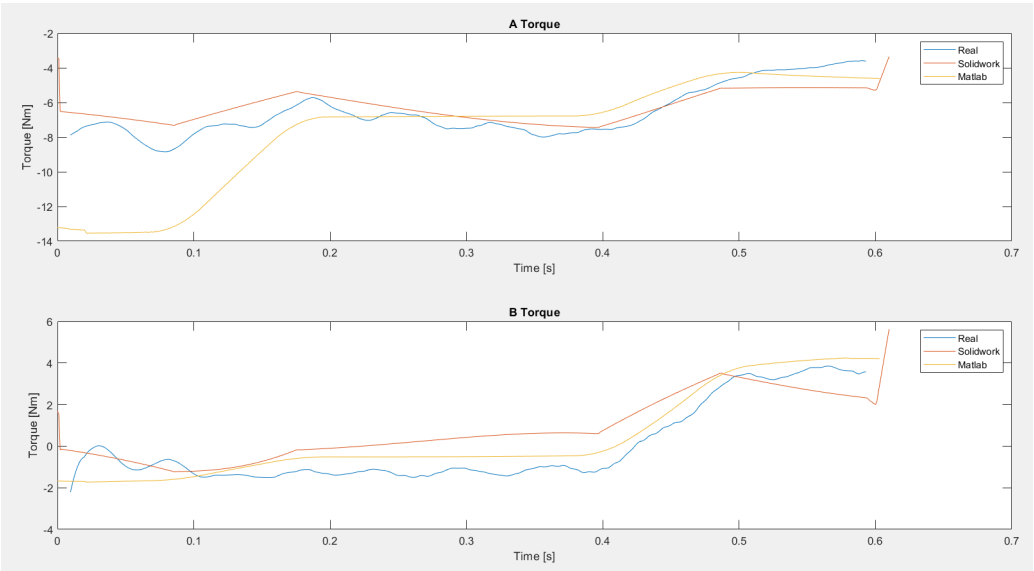

Figure 4.12: Motors Torque Comparison

4.2.3 Upshot

To do this test, we clean the data at different times. It is due that the simulation starts from the collected value and not from an internally defined function. The torques obtained are not the same but have a similar trend.

The simulation is run various times because errors were found in the result of every simulation. Every time I correct some error, the trend becomes more similar, but in the end, the result with the correct equation is the one reported in Figure 4.12.

The main difference can be observed at the beginning of the Matlab torque C_a. The Matlab-simulated dynamic system is very different from the real one. This conclusion may be due to the different initial conditions. The initial condition is changed from zero to a value similar to the real one. The results did not change as much as expected.

The test helps to find errors in the dynamics equation. It also shows that the torques obtained in Solidworks have a trend similar to those obtained from actual data.

We decided to do another test to simulate a trajectory with a movement defined directly in Solidworks. These passages are explained in Test 3.

4.3 Test3 - New trajectory

4.3.1 Objective

This last test is done to verify if we can obtain smooth torque curves from Solidworks. This goal is reached by defining as input a mathematical function internally described by Solidworks.

The main goal of this test is to understand all the limitations of motion. The passages are the same as those used in the office to analyze a new trajectory.

Kinematic data will also be used to simulate the Matlab model. They are analyzed to find whether there is a similar trend between the two models.

4.3.2 Procedure

First, a new trajectory must be described. The trajectory is simulated at a different height. The end effector will remain the same.

The trajectory is a straight path from A to B; the start and end values of the straight line are reported in the following table.

	X mm	**Y mm**
A	500	110
B	1200	110

Table 4.3: Test 3 Path

The path mate motor is defined with a segment function builder. The segment is 700mm long and must be completed in 0.5 seconds. The type of segment is defined as a 3-4-5 polynomial function. The characteristics of this function are the following: the velocity and acceleration are zero at the start and end of the segment; the jerk is not zero at the start and end, but it is finite; the segment is symmetric about its midpoint.

This curve is chosen because it is similar to the one we have in the actual simulation. The displacement, velocity, and acceleration can be seen in the image on the following page.

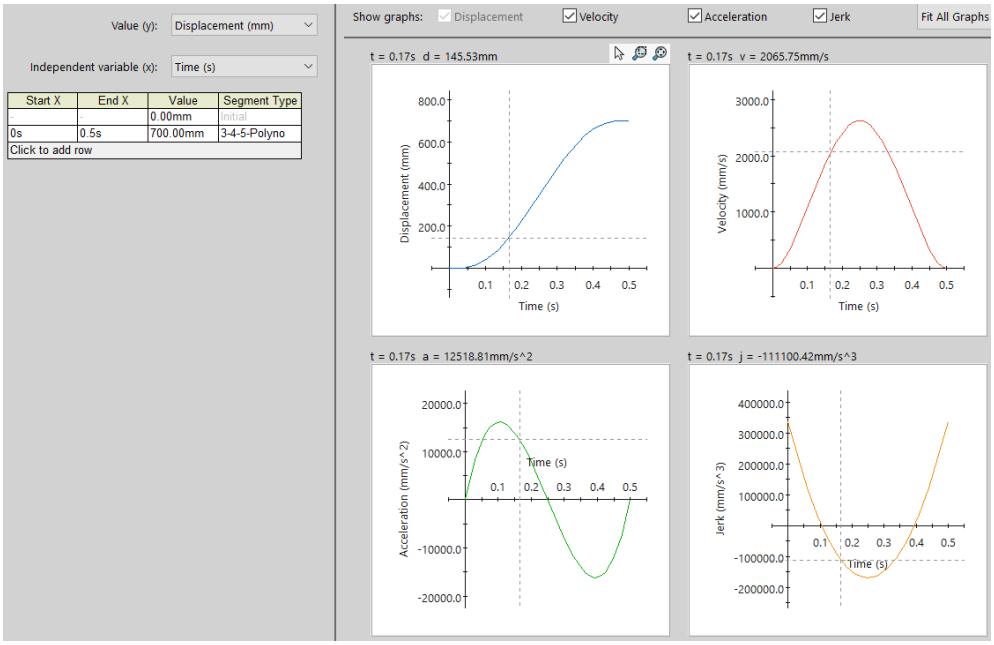

Figure 4.13: Function builder

Then the data from the shaft can be extracted and used in the direct dynamic study. Having done all the steps to upload the input data, we found that the path is wrong. The discrepancy is visible in the first simulation.

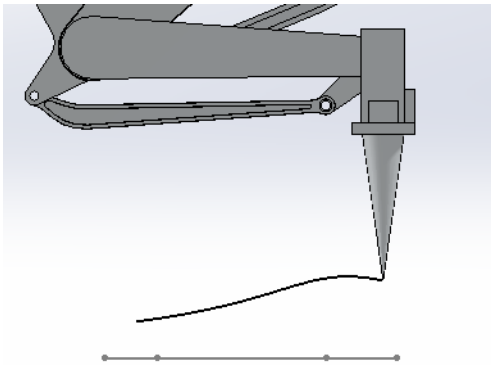

Figure 4.14: Path discrepancy

Another helpful tip was found to solve this problem: Instead of taking the displacement from the shaft is better to take them from the **hinge**. The data will be from different orientations, but if the motor is set accurately, the results are correct.

The two different ways to take data are shown below. On the right are the data from the hinge, and on the left, data from the shaft. It can be immediately seen that *Hinge B* is opposite to *Shaft B*.

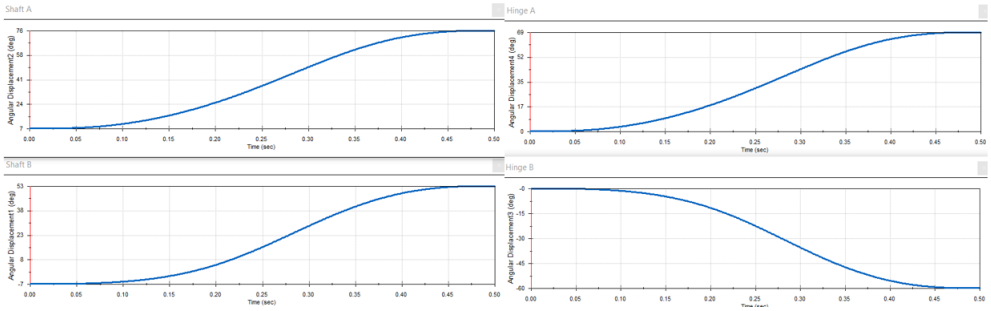

Figure 4.15: Shaft and Hinge

To correctly upload the data to *motor B*, we must set the correct direction of rotation. In this case, we must set it clockwise, not the default choice. Simulating the motion, we obtain a correct path followed by the end effector.

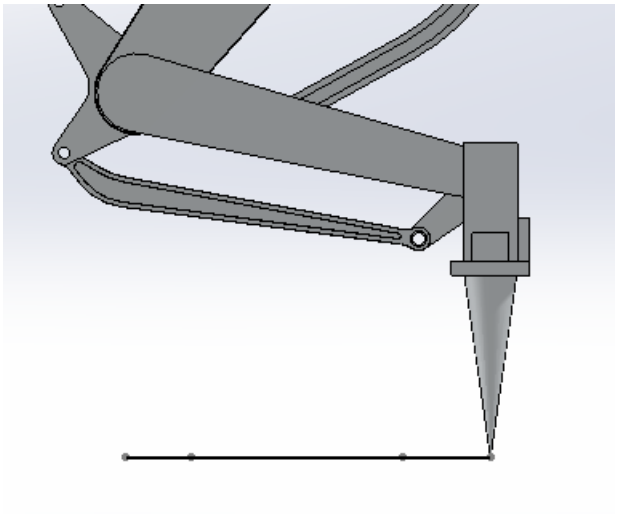

Figure 4.16: Correct Path

The torque can be extracted from both motors. Their values are exported from the following graphs.

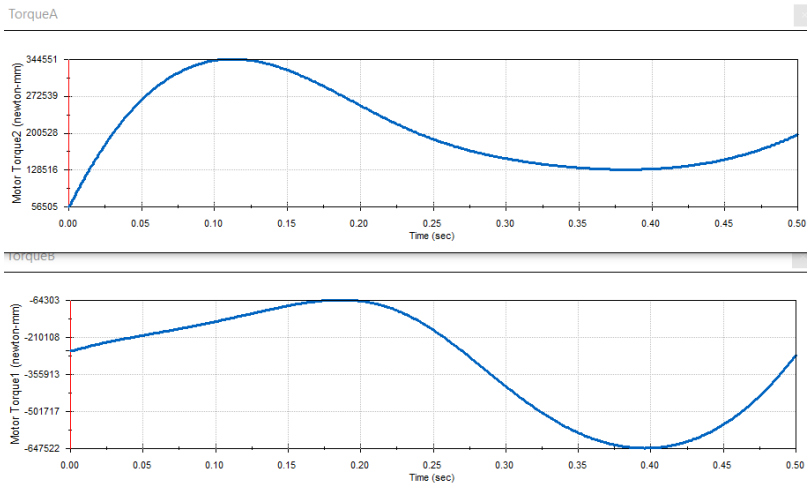

Figure 4.17: Correct Torque

To test the trend, I also decided to extract the torque curve from the simulation reported in Figure 4.14. I know the torques are different, but the trend of torque B must be similar.

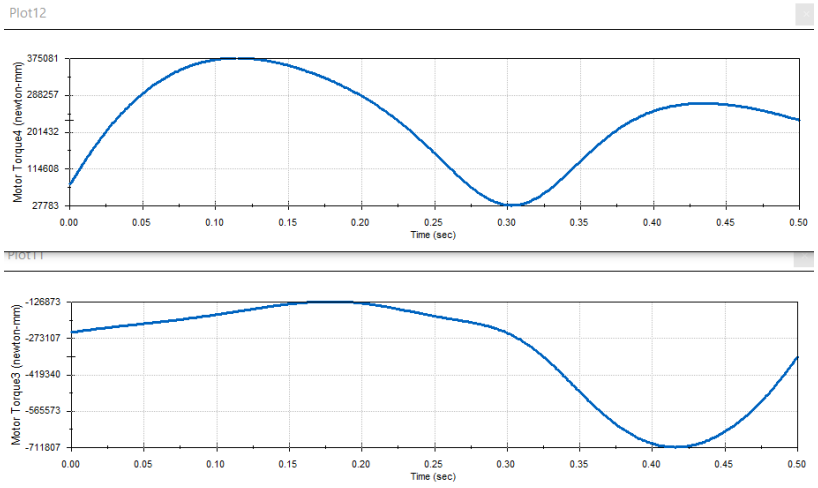

Figure 4.18: Wrong Trajectory

The simile of torque B is because the movement is on a horizontal line in both cases. The CSV file is exported and reported in Matlab to analyze the torque.

4.3.3 Upshot

The test helps us understand other errors we can have during the simulation. The setup and the step order are essential to be followed correctly.

The trajectory must be defined as efficiently as possible for a correct analysis. From Matlab simulation, we again obtain a similar torque trend. Both models' maximum and minimum values are simultaneously but with different values.

Chapter 5

Conclusion

SACMI has as its objective the creation of a manipulator model to study the torque needed to follow a defined trajectory. To realize this model, the software available in the office are analyzed. The available software that can be used are Solidworks and Excel.

After a better analysis of Solidworks, it was found that Motion was an available add-on for the study. It allows us to simulate movements and collect data about kinematics, forces and torque.

To realize this model, I started from a Solidworks assembly of the TLP. The components of this assembly are simplified and coupled together to allow movements.

At the same time, the dynamic study of the TLP is devoted to creating a Matlab model to confirm the simulation. The two models are compared reciprocally and with real data to see if they have a similar trend.

The tests helped to find different errors that can be made during the setup of the motion analysis. The main errors or best practices found are reported here.

- Use simple paths and only one-way movements. A round-trip motion must be divided into two different studies: gone and return;
- Prefer to use the predefined mathematics function available in Solidworks for displacement and velocity input;
- To avoid calculation errors, it is preferred not to overwrite the Inertia Moments; only the mass and center of mass will be overwritten for the end effector. The TLP model is created with the correct mass.
- Matlab model has a trend similar to the one obtained by Solidworks and the data from real movement. A different beginning for C_a is noticed due to different initial conditions. Changed them, the result did not change much. Also for C_b it is possible to appreciate different initial conditions, but they are less evident than for C_a.

- For every simulation, it is preferred to clean the motion result to avoid superposition of error. It is also possible to have calculation errors because we have a complex graphical simulation,

It can be said that the model can be used to analyze the torque needed to obtain the desired path. The process is not direct but must be followed in a precisely ordered order, step by step. The output torque must be scaled and reported to the torque at the motor shaft.

One possible improvement recommended by one office is to try to recreate the model in Solidworks 2007. This old version is more stable than the newer version for the motion. However, it is not present on the workstation available in the office. So, to create this new model, we first have to downgrade a Solidworks version on an old workstation, recreate the model from zero, and perform the analysis. They told us that a better solution is not a guarantee.

Further improvements can be made to connect Solidworks directly to Excel using queries written in Visual Basic for Applications (VBA). Those queries can automatically run the processes needed to simulate the motion step by step.

Another tried simplification was the remotion of secondary bodies. These bodies compose the kinematics that forces the end-effector to stay horizontal. This simplification does not change the step and does not accelerate the simulation, instead of adding a non-utile simplification that can be avoided.

We can conclude that the models have a similar trend. The more accurate one respect the actual data is the simulation accomplished in Solidworks. It is the one that will be used by the office.

Appendix A

Position and Angles

```
clc
clear all
close all

for i=1:1675
    for k=1:860

        Ei=[i, k];

        x0=[0,0,0];
        x=fsolve(@(x) Sys_eq(x, Ei), x0);

        for j=1:4
           MATR(i,k,j)=x(j);
        end
    end
end
```

```
function F = Sys_eq(x, Ei)

A1=260; A2=750; B1=550; L1=260; L2=440; E=185.5;

P=700; O=180; N=230; M=550;

L=L1+L2;

Ax=1096.2;   Ay=1283.41;
Bx=850;      By=1240;
Xx=702.16;   Xy=1416.19;

Ex=Ei(1);    Ey=Ei(2);

     F(1)=Ex-Ax-L2*sin(x(4))+A2*cos(x(2))-A1*sin(x(1));
     F(2)=Ex-Bx-L*sin(x(4))+B1*cos(x(3));
     F(3)=Ey+E-By+L*cos(x(4))-B1*sin(x(3));
     F(4)=Ey+E-Ay+L2*cos(x(4))-A2*sin(x(2))+A1*cos(x(1));

end
```

Appendix B

Working Area

```
clc
clear all
close all

for i=1:1675
    for k=1:860
            %ALFA           [-35; 150]
            %GAMMA          [-110; 40]
            %DELTA+GAMMA    [-24; 50]

        if (MAT(i,k,1)>deg2rad(-35) && MAT(i,k,1)<deg2rad(150)) &&...
           (MAT(i,k,3)>deg2rad(-110) && MAT(i,k,3)<deg2rad(40)) &&...
           ((MAT(i,k,4)+MAT(i,k,3))>deg2rad(-24) && (MAT(i,k,4)+MAT(i
    ,k,3))<deg2rad(50))

                %Reachable
                Working_Area(i,k)=1;
        else
                %Non reachable
                Working_Area(i,k)=10;
        end
    end
end

figure(1)
contourf(Working_Area',1)
axis equal
```

Appendix C

Real Data

```
M = readmatrix('Clean.txt');
i=1; r=1; c=1;

while r <= 1468605
    while c <= 27
          if ~ismissing(M(r,c))==0
              r=r+1;
          else
              Dati(i,c) = M(r,c);
              c=c+1;
          end
                if c>27
                    break
                end
    end


                if r==1468605
                    break
                end
    c=1; i=i+1;
end

%writematrix(Dati, 'DatiT.xlsx')
```

Appendix D

Main Matlab Model

```
%% Verification using a known straight trajectory.

EXi=407;        %mm initial position
EXo=1373;       %mm final position
EY=282;         %mm trajectory position Y axes

%% Position, velocity and acceleration Xe

P = csvread('RealData.CSV');     %read CSV file

Te  = P(:,1)';                   %time
Xe  = P(:,2)'+EXi;               %position
dXe = P(:,3)';                   %velocity

ddXe = diff(dXe)./diff(Te);      %acceleration
ddXe =[ddXe(1) ddXe];

Nsamp=length(P);          %number of sample

%% Angular position
for i=1:Nsamp

    Ei=[Xe(i), EY];
    x0=[0,0,0,0];

    x=fsolve(@(x) IN_Pos(x, Ei), x0);

    PVA(i,1,1) =   deg2rad(270)+x(1);   %Alfa
    PVA(i,2,1) =   deg2rad(180)-x(2);   %Beta
    PVA(i,3,1) =   deg2rad(180)-x(3);   %Gamma
    PVA(i,4,1) =   deg2rad(270)+x(4);   %Delta
```

```
32
33 end
34
35 %% Velocity and acceleration angular profile
36
37 for k=1:4
38
39     ANG=PVA(:,k,1)';
40
41     dANG = diff(ANG)./diff(Te);
42     dANG=[dANG(1) dANG];
43
44     ddANG=diff(dANG)./diff(Te);
45
46     PVA(:,k,2)=[dANG]';
47     PVA(:,k,3)=[ddANG(1) ddANG]';
48 end
49
50 PVA(:,5,1)=PVA(:,4,1); PVA(:,5,2)=PVA(:,4,2); PVA(:,5,3)=PVA(:,4,3);
51 PVA(:,6,1)=PVA(:,3,1); PVA(:,6,2)=PVA(:,3,2); PVA(:,6,3)=PVA(:,3,3);
52
53 %% Dynamic solver for every instant
54
55 for i=1:Nsamp
56
57     C = Dinamica(i, PVA, ddXe);
58
59     %from mm^2 to m^2
60     Ca(i)=C(1)*10e-6;
61     Cb(i)=C(2)*10e-6;
62
63 end
```

Appendix E

Function DINAMICA

```
function C = Dinamica(i, PVA, ddXe)

[g, Tb, dTb, ddTb, Bg, M1, BD, IZ1, Td, dTd, ddTd, M2, Dg, L, L1, IZ2
    , Tc, dTc, ddTc, M3, Cg, Alph, CF, mem, IZ3, Ta, dTa, ddTa, M4, Ag
    , Heth, AC, Dx, IZ4, M5, Gg, Beth, EG, Dx5, Kx, Ky, IZ5, Tw, dTw,
    ddTw, M6, Wg, gamal, WK, IZ6, M7, Dg7, He, Dx7, wd, wy, IZ7, Tx,
    dTx, ddTx, M8, Xg, waw, xy, IZ8, ddx, MPinza, GE, IZE] =
    Dimensions(PVA, i, ddXe);

Tex  =  MPinza*ddx*Gg;
Tey  = -MPinza*g;
ddTg = (Tex*GE)/(-IZE+MPinza*Gg^2);

     x0=zeros(1, 14);
     x=fsolve(@(x) Din(x, i, PVA, Tex, Tey, ddXe), x0);

Tfx= x(5);   Tfy= x(6);

Tdx= x(1);   Tdy= x(2);

Tcx=-Tfx+M3*dTc^2*Cg*cos(mem)+M3*ddTc*Cg*sin(mem);
Tcy=-Tfy+M3*g+M3*dTc^2*Cg*sin(mem)-M3*ddTc*Cg*cos(mem);

Ca=ddTa*IZ4+M4*ddTa*Ag^2+Tcx*AC*cos(Ta-deg2rad(270))+Tcy*AC*sin(Ta-
    deg2rad(270))+M4*g*Dx;
Cb=ddTb*IZ1+M1*ddTb*Bg^2+Tdy*BD*cos(Tb-deg2rad(180))-Tdx*BD*sin(Tb-
    deg2rad(180))-M1*g*Bg*cos(Tb-deg2rad(180));

C = [Ca Cb];
end
```

Appendix F

Function DIN

```
function F = Din(x, i, PVA, Tex, Tey, ddXe)

[g, Tb, dTb, ddTb, Bg, M1, BD, IZ1, Td, dTd, ddTd, M2, Dg, L, L1, IZ2
    , Tc, dTc, ddTc, M3, Cg, Alph, CF, mem, IZ3, Ta, dTa, ddTa, M4, Ag
    , Heth, AC, Dx, IZ4, M5, Gg, Beth, EG, Dx5, Kx, Ky, IZ5, Tw, dTw,
    ddTw, M6, Wg, gamal, WK, IZ6, M7, Dg7, He, Dx7, wd, wy, IZ7, Tx,
    dTx, ddTx, M8, Xg, waw, xy, IZ8, ddx, MPinza, GE, IZE] =
    Dimensions(PVA, i, ddXe);

F(1) = -x(1)-x(3)-x(5)-x(7)-M2*ddTd*Dg*cos(Td-deg2rad(180))-M2*dTd^2*
    Dg*sin(Td-deg2rad(180));
F(2) = -x(2)-x(4)-x(6)-x(8)+M2*ddTd*Dg*sin(Td-deg2rad(180))-M2*dTd^2*
    Dg*cos(Td-deg2rad(180))-M2*g;
F(3) = +x(8)*L*cos(Td-deg2rad(180))-x(7)*L*sin(Td-deg2rad(180))+x(6)*
    L1*cos(Td-deg2rad(180))-x(5)*L1*sin(Td-deg2rad(180))+M2*g*Dg*cos(
    Td-deg2rad(180))-M2*ddTd*Dg^2-IZ2*ddTd;

F(4) = x(9)+x(7)+Tex;
F(5) = x(10)+x(8)+Tey-M5*g;
F(6) = Tex*EG+M5*g*Dx5+x(9)*Kx-x(10)*Ky;

F(7) = x(11)+x(3)+x(13);
F(8) = x(12)+x(4)+x(14)-M7*g;
F(9) = M7*g*Dx7-x(12)*wd-x(13)*wy;

F(10) = -IZ3*ddTc-x(6)*CF*cos(deg2rad(180)-Tc)-x(5)*CF*sin(deg2rad
    (180)-Tc)+M3*g*Cg*cos(Alph-(deg2rad(180)-Tc))-M3*ddTc*Cg^2;

F(11) = -x(11)-x(9)-M6*ddTw*Wg*cos(gamal-(Tw-deg2rad(270)))-M6*dTw^2*
    Wg*sin(gamal-(Tw-deg2rad(270)));
```

```
20 F(12) = -x(12)-x(10)+M6*ddTw*Wg*sin(gamal-(Tw-deg2rad(270)))-M6*dTw
       ^2*Wg*cos(gamal-(Tw-deg2rad(270)))-M6*g;
21 F(13) = -ddTw*IZ6-x(9)*WK*cos((Tw-deg2rad(270)))-x(10)*WK*sin((Tw-
       deg2rad(270)))+M6*g*Wg*sin(gamal-(Tw-deg2rad(270)))-M6*ddTw*Wg^2;
22
23 F(14) = -ddTx*IZ8+M8*g*Xg*cos(waw+(deg2rad(180)-Tx))-M8*ddTx*Xg^2+x
       (14)*xy*cos((deg2rad(180)-Tx))+x(13)*xy*sin((deg2rad(180)-Tx));
24
25 end
```

Bibliography

[1] «The company». In: (). URL: https://www.sacmi.it/it-IT/packaging/news/9388/SACMI-Packaging-Chocolate-costituisce-la-nuova-B-U-Tray-Forming (cit. on p. 1).

[2] «Cambrige». In: (). URL: https://dictionary.cambridge.org/it/dizionario/inglese/packaging (cit. on p. 2).

[3] «History». In: (). URL: https://ohioline.osu.edu/factsheet/cdfs-133#:~:text=The%20%EF%AC%81rst%20commercial%20cardboard%20box,and%20boxes%20used%20for%20trade. (cit. on p. 2).

[4] «Modern». In: (). URL: https://www.entrepreneur.com/encyclopedia/packaging#:~:text=Packaging%20Definition%3A,marketable%20and%20keep%20it%20clean (cit. on p. 2).

[5] «Primary». In: (). URL: https://www.emballagecartier.com/en/article/primary-secondary-and-tertiary-packaging-whats-the-difference/#:~:text=Primary%20packaging%20is%20the%20packaging,finished%20product%2C%20particularly%20against%20contamination. (cit. on p. 3).

[6] «Seconday». In: (). URL: https://www.alcaminow.com/blog/the-importance-of-secondary-packaging#:~:text=Secondary%20packaging%20is%20the%20exterior,handling%2C%20storage%2C%20and%20distribution. (cit. on p. 4).

[7] «Tertiary». In: (). URL: https://www.airseacontainers.com/blog/primary-secondary-tertiary-packaging-guide-to-3-levels-of-packaging (cit. on p. 4).

[8] «HELP». In: (). URL: https://help.solidworks.com/2021/english/SolidWorks/sldworks/r_welcome_sw_online_help.htm?id=a3065a38347d43b2b21ee1ea417733ca#Pg0 (cit. on p. 7).