Università degli Studi di Ferrara

DOTTORATO DI RICERCA IN

"SCIENZE DELL'INGEGNERIA"

CICLO XXXIII

COORDINATORE Prof. TRILLO STEFANO

OPTIMIZING THE PERFORMANCE OF THE INDUSTRIAL ROBOT CTFLEX IN PACKAGING AUTOMATION SYSTEMS

Settore Scientifico Disciplinare ING-INF/04

Dottorando

Dott. HMADI MOHAMMAD

Tutore

Prof. BONFE' MARCELLO

Anni 2017/2020

ABSTRACT

This experimental Ph.D. research, developed within the CT Pack company, manufacturer of machines and plants for food packaging, aims at improving the performance (i.e. in terms of products per minute) of the operations of picking and placing of products in a packaging line. In particular, the objective is to develop a new Delta robot, fully realized inside the company, with unique mechanical features in terms of payload capacity with using the latest technologies for industrial automation and control.

This research illustrates the development of a novel parallel Delta robot. The kinematic structure known as Delta robot is one of the most important inventions in the industrial machinery sector, mainly used in the packaging industry. The robot specifically developed in CT Pack Company is called CTFLEX. Its development consists of choosing the proper control hardware and creating the programming software to configure new characteristics as mechanically performed; the software has been chosen to be the same used in PLC controllers of the machines, obtaining a robot well integrated into the same line. This research will further demonstrate the uses of this robot, the steps during software development and the optimization of its performance.

Table of Contents

1. Introduction

This research results from three years of work and experimental analysis dedicated to studying a new technology introduced in the field of robotics.

The host company is called CT Pack S.R.L based in Italy, specialized in the food packaging sector. It is an expert in robotic systems for handling and canning different products at high speed based on industrial objectives and strategies.

The Research & Development departments keep alive the interest in working closely with the Faculty of Engineering of the University of Ferrara. Among the company's various production sectors, the company decided to deepen the secondary packaging machines. These machines are used in the packaging of ice creams, biscuits, snacks, bars, tablets, and chocolates.

Starting from the production line infeed, the product arriving from the production line continues its path through this type of system to be picked up and deposited by the delta robot in the appropriate destination. It is easy to notice that most of the machine movements achieved through electric motors, especially for the delta robot's high performance, which is the fundamental point for the entire thesis.

The research focuses on developing a new delta robot, see Figure 1, with the latest features necessary to be integrated into the new types of the packaging machine to have economic and higher performance advantages over other delta robots in the market. The thesis also discusses optimizing other commercial robots' performance for the same uses. The company decided to develop the robot using Rockwell Automation products, more precisely Allen Bradley hardware controllers with Studio 5000 programming software, due to their high performance and efficiency and the long experience of using them in the CT Pack machines.

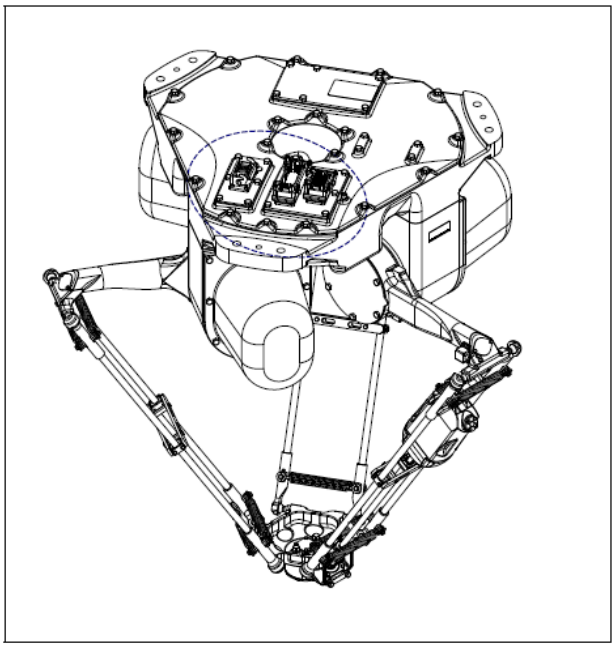


Figure 1 Parallel Delta Robot

2. Parallel Robot

There are essentially two types of robot manipulators: serial and parallel. Serial manipulators consist of several links connected in series to one another to form a kinematic chain. Each joint of the kinematic chain is usually actuated. This type of structure is known as an open chained mechanism. On the other hand, parallel manipulators consist of several kinematic chains connected in parallel to one another. The kinematic chains work in union to move a common point. This common point usually consists of a manipulator that performs a particular task. For the three degrees of freedom (3 DOF) parallel DELTA robot system described in this thesis, the common point will also be referred to as the end effector. Since the kinematic chains are eventually connected to a common point, a parallel manipulator is considered a closed chained mechanism. In parallel manipulators, the actuators are usually located at the base or close to the system's base, which is in stark contrast to serial manipulators that have actuators at every joint. The advantages of this type of configuration include the fact that it could achieve a higher load capacity due to the overall system's mass decrease. It can produce high accelerations at the end effector and high mechanical stiffness to weight ratio [1].

The disadvantages of this type of configuration include the fact that the dynamic model is naturally complicated. Many instances of singularities must be mapped out and avoided to maintain control of the system. Parallel robots come in a wide variety of designs and applications ranging from the Stewart platform or Hexapod Parallel Robot shown in Fig. 2.1, used in aircraft motion simulators, and Delta robot used in packaging plants. There cannot be a conclusive result as to which controller best suits all parallel robots' functionality. Therefore, it is logical to experiment with various control techniques to observe which controller would garner the most satisfactory results based on a specific mechanical system.

This system's dynamic model parameters are derived in detail, followed by the derivation of the mechanical model's inverse kinematics. The non-singular region is then defined based on the results obtained in the inverse kinematics. It is essential to map out the non-singular area since it is the only location where the parallel robot can operate under stable conditions. If the parallel robot were to enter a singular region, it would render the controller ineffective and cause the entire system to become unstable. It is impossible to adequately design any controllers for the parallel robot without a clear understanding of the dynamic and mechanical models' inverse kinematics.

In recent years the number of studies and applications of parallel robots has increased. One of the most popular applications is in industry packaging. The above is due to ease of construction and lightness of structure, and the high accelerations obtained by these devices.

Unlike the serial-type robot manipulators, which only have an open-loop kinematic chain, the parallel configuration allows for a distribution of payload among their two or more closed-loop kinematic chains. To illustrate this point, consider Fig.2.1 shows a parallel-architecture robot used for object loading and unloading. Fig.2.2 shows a SCARA-type serial-architecture robot. By comparing the images, it is easy to appreciate the difference between the two types of architecture. In the case of the serial manipulator, it required greater robustness, as each link carries the weight of the successive links and the motors and payload, which creates a cantilever effect in each link and, as a result, a more significant deformation overall.

In contrast, in the parallel architecture, the actuators are fixed to the manipulator base so that the kinematic chains do not support the motors' weight. Also, the distributed payload is among the kinematic chains that con-form the manipulator, which results in thinner and lighter kinematic chains. It increases the manipulator's payload capacity relative to its total mass.

(A) *(B)*

Figure 2.1, (A) Hexapod Parallel Robot, (B)CODIAN Robotics Parallel robot

Figure 2.2 SCARA-type serial-architecture robot

3. Theoretical Foundation

The previous chapter has introduced the advantages and disadvantages of the parallel robot and compared their performance with serial type robots. This chapter study the kinematics of 3 DOF Parallel DELTA robot.

3.1 DELTA Type Parallel Robot

The well-known Delta robot structure was proposed by R. Clavel in [2]. Fig. 2.3 shows the main components of this robot, which consists of three or four closed-loop kinematic chains. The robot has three degrees of freedom.

The parallelograms ensure the constant orientation between the fixed and the mobile platform, allowing only translation movements of the latter. The end effector of the manipulator is located on the mobile platform [3].

Parallel Robot can move products in a three-dimensional Cartesian coordinate system.

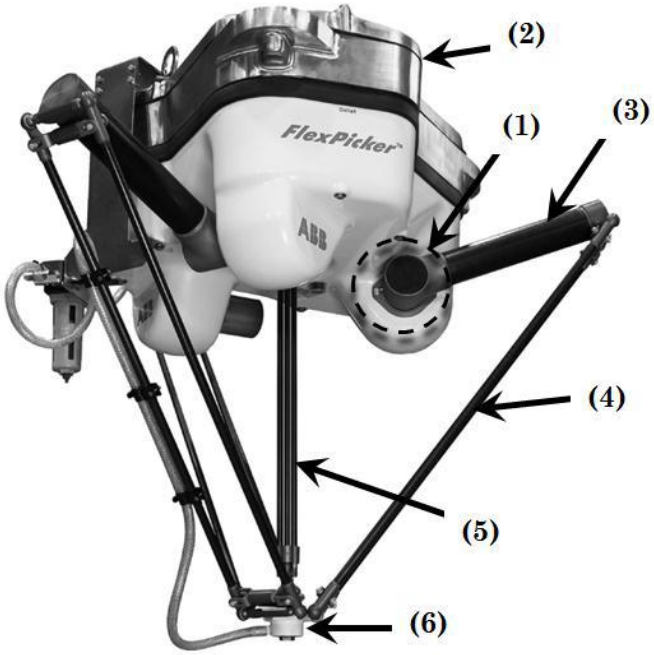


Figure 3.1 Parallel DELTA Robot Components

The combination of the constrained motion of the three arms connecting the traveling plate to the base plate ensues in a resulting 3 translator degrees of freedom (DOF). As an option, with a rotating axis at the Tool Center Point (TCP), four DOF are possible.
The Robot consists of, consider Fig.2.3:

1. Three Actuators.
2. Base plate.
3. Upper robot arm.
4. Lower robot arm (Forearm).
5. Rotation arm (optional, 4-DOF).
6. Travelling plate, TCP.

The upper robot arms are mounted direct to the actuators to guarantee high stability. And the three actuators are rigidly mounted on the base plate with 120° in between. Each of the three Lower robot arms consists of one bar, which connects the upper arm with the travelling Plate via ball joints Lower frictional forces result from this. The wear reduces respectively as a result. To measure each motor shaft, angle a Quadrature Optical encoder is used (in the asynchrony motor of Allen Bradley there are installed internally an encoder). A rotational axes is available for the robot mechanics as an option. The actuator for this axis is then mounted on the upper side of the robot base plate (which have an encoder). The bar is connected directly to the tool and ensures for an additional rotation motion [4].

3.2 Inverse Kinematics

The purpose of determining the inverse kinematics of this parallel robot is to accurately model the angle produced at each joint at a specific location of the end-effector (Travelling plate). This is advantageous for two main reasons; the first being that it is relatively simple to define any reasonable trajectory for the end effector to traverse and secondly, it can track different trajectories in a non-singular region [5].

The constrained three degrees of freedom system shown in Fig. 2.4 will also be applicable in this section. It should be noted that the parameters of the overall system are known, which include: the range of the desired angles for θ_1, θ_2 and θ_3 respectively, the overall length of each upper link La and the overall length of each lower link Lb. the desired location of the end effector in the x and y axis respectively and the horizon distance between the two motor shafts (c).

The problem of the Inverse kinematics solution is to find the actuators states θ_1, θ_2, θ_3, known the end-effector position (x, y, z). To find the inverse kinematics solution let us refer to Fig.2.5 Also, let consider the origin of the reference system fixed on the platform and the axes such as depicted in Fig. 2.6 Note that the parallelogram has been considered as a single link (lb).

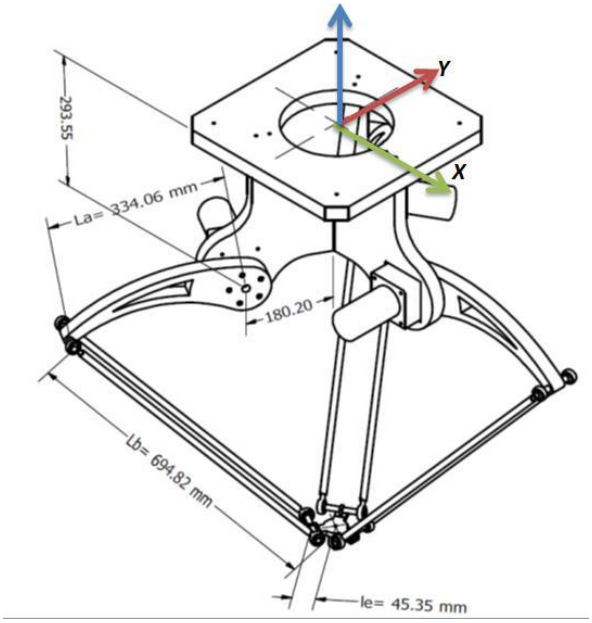


Figure 3.2 Delta Parallel Robot Side View Designed on Autodesk Inventor

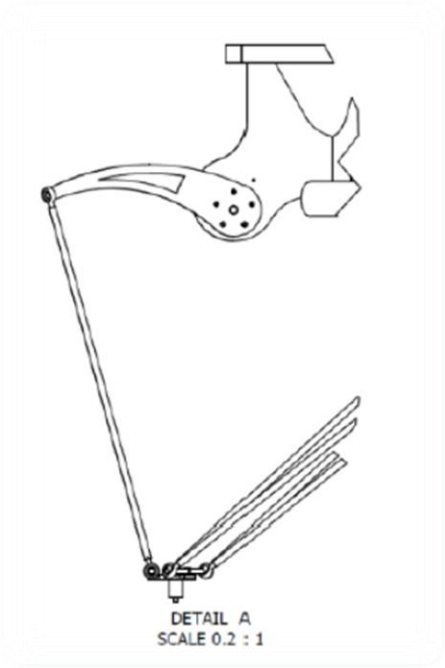


Figure 3.3 One link side view of DELTA Parallel Robot

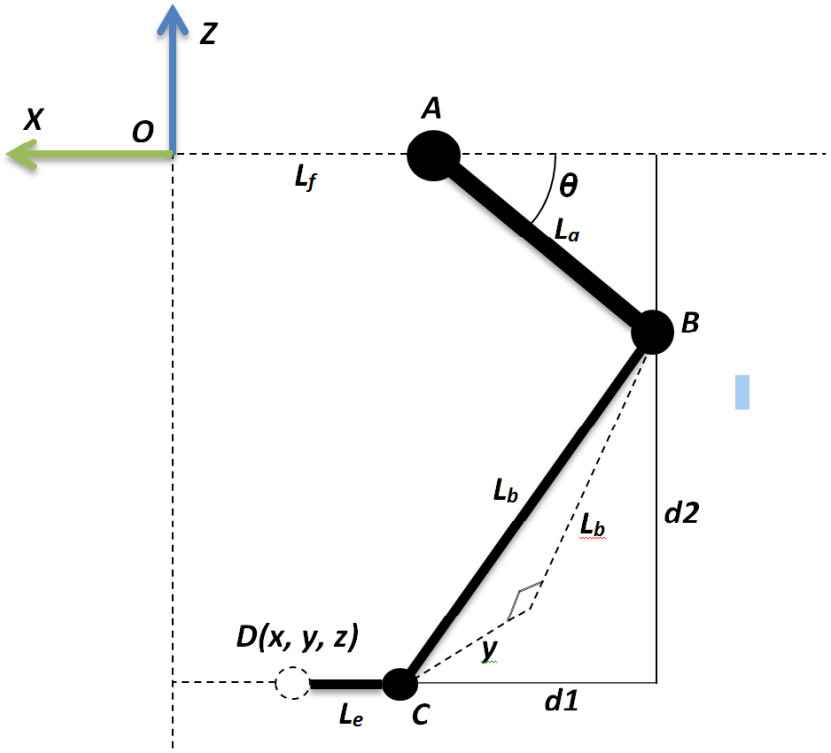


Figure 3.4 First kinematics chain, XZ plane projection

The analysis begins considering each kinematics chain separately, for the first kinematics chain; shown in Fig.2.5, we make a projection to the X-Z plane, which yields a vector closed loop as shown in Fig.2.6.

From Fig. 2.6, we have:

$$d_1^2 + d_2^2 = l_b'^2 \tag{2.1}$$

Where,

$$l_b'^2 = l_b^2 - y^2 \tag{2.2}$$

In addition, we have that:

$$OA + l_a\, cos(\theta_1) = -x + DC + d_1$$

Solving for d1, yields:

$$d = T + l_a\, cos(\theta_1) \tag{2.3}$$

Where,

$$T = OA + x - DC$$

Also, from the geometry of Fig. 2.7 we have,

$$d_2 = z + l_a\, sin(\theta_1) \tag{2.4}$$

Substituting (2.2), (2.3) and (2.4) in (2.1) and simplifying, we obtain:

$$2T_1 l_a\, cos(\theta_1) + 2z l_a\, sin(\theta_1) = K \tag{2.5}$$

With,

$$K = l_a^2 - l_b^2 - x_0^2 - z^2 - T^2$$

Substituting in (2.5) the trigonometric identities,

$$cos(\theta_1) = \frac{1-t^2}{1+t^1} \ , \ sin(\theta_1) = \frac{2t}{1+t^1} \ , where, t=tan\left(\frac{\theta_1}{2}\right)$$

We obtain:

$$e_1 t^2 + e_2 t + e_3 = 0 \qquad \textbf{(2.6)}$$

Where:

$$e_1 = 2Tl_a + Kl_a$$

$$e_2 = -4zl_a$$

$$e_3 = -2Tl_a + K$$

Solving (2.6) for t yields,

$$\theta_1 = 2\, tan^{-1} \frac{-e_2 \pm \sqrt{e_2^2 - 4e_1 e_3}}{2e_1} \qquad \textbf{(2.7)}$$

From the previous equations, we can conclude that,

$$\theta_1 = f(x, y, z) \qquad \textbf{(2.8)}$$

Following the same procedure, the others two kinematics chains configurations can be solved. We can take advantage of the symmetry of Delta Robot and consider the fact that each kinematics chain is rotated 120 degree relative to each other. We could take the base the first kinematics chain and multiply it by the rotation matrix (120° for θ_2 and 140° for θ_3) and then apply the process used to solve the first kinematics chain. Once followed the procedure described previously, the values of θ_2 and θ_3 can be found. In general, there are a total of eight possible robot postures corresponding to a given end-effector location [6].

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \begin{bmatrix} cos(\theta_1) & sin(\theta_1) & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \textbf{(2.9)}$$

Where,

$$x' = cos(\alpha)\, x - sin(\alpha) y$$

$$y' = sin(\alpha)\, x - cos(\alpha) y$$

$$z' = z \qquad \textbf{(2.10)}$$

Where α is the angle of rotation about z axis, From Eq. (2.10) yields:

$$\theta_{2,3} = f(x', y', z') \qquad \textbf{(2.11)}$$

Hence, there are generally two solutions of θ_1 and therefore two configurations of the kinematics chain Fig. 1.7 corresponding to each end-effector location. When Eq. 2.7 yields a double root, the two links of the kinematics chain are in a fully stretched-out or folded-back configuration named singular configuration. When Eq. 2.7 yields no real solution, the specified end-effector location is not reachable. Despite of the two possible solutions, only the negative root has to be taken because the positive one could cause interference between the elements of the robot as depicted in Fig 1.7.

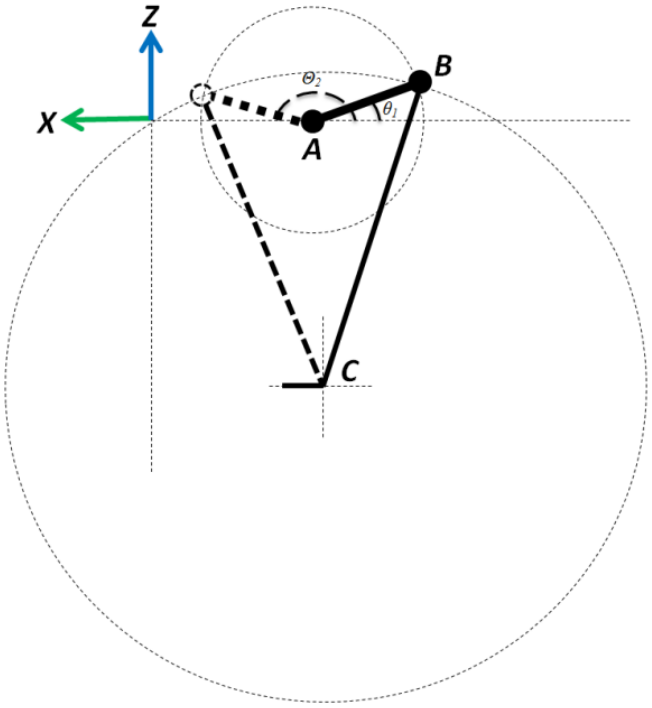


Figure 3.5 Two possible configurations of the kinematics chain due to θ1

The inverse kinematics solution is tested for special cases by examining Eq. 2.7: If $e_2^2 4e_1e_3 > 0$, then the circle swept by vector AB intersects the sphere swept by vector BC in two locations. If $e_2^2 4e_1e_3 = 0$, then the circle and sphere are tangent, and the manipulator is in a singular position. If $e_2^2 4e_1e_3 < 0$, then the circle and the sphere

do not intersect and there are no real solutions. If $e_1 = e_2 = e_3 = 0$, then the circle lies on the sphere, and there are infinite number of solutions.

3.3 Forward Kinematics

The forward kinematics also called the direct kinematics of a parallel manipulator determines the *(x, y, and z)* position of the travel plate in base-frame, given the configuration of each angle θ_i of the actuated revolute joints, see Fig.2.8.

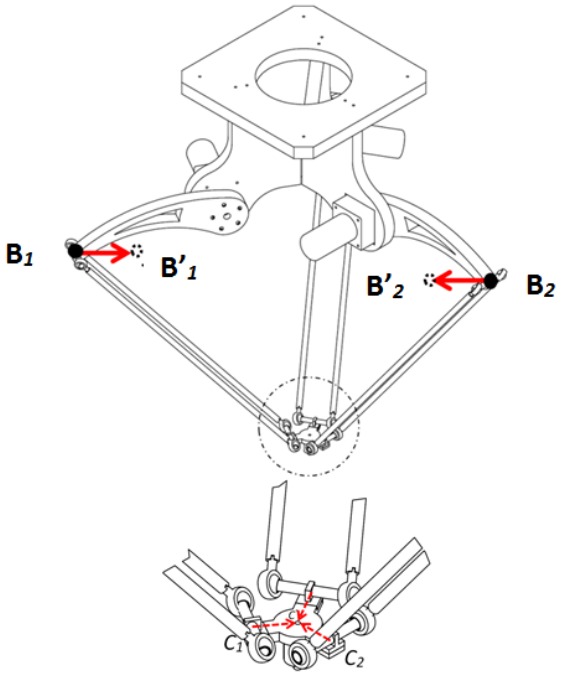


Figure 3.6 Configuration chosen for direct kinematics analysis

Consider three spheres each with the center at the elbow B_i of each robot arm chain, and with the forearm's lengths l_b as radius. The forward kinematic model for a Parallel Delta Robot can then be calculated with help of the intersection between these three spheres. When visualizing these three spheres they will intersect at two places.

One intersection point where z is positive and one intersection point where z coordinate is negative. Based on the base frame {R} where z -axis is positive upwards the TCP will be the intersection point when z is negative. Fig. 2.9 shows the intersection between three spheres. Where two spheres intersect in a circle and then the third sphere intersects this circle at two places.

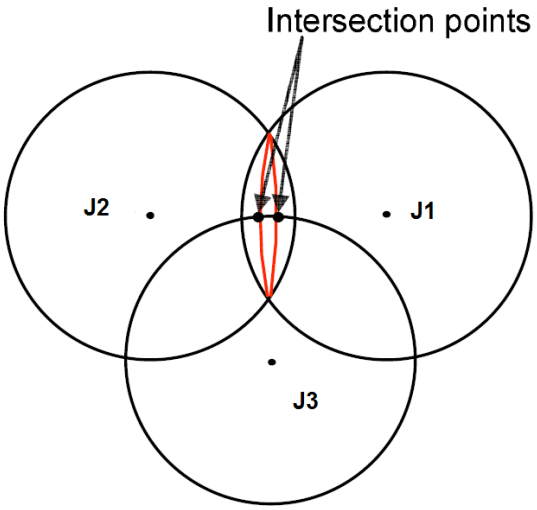


Figure 3.7 Two spheres intersect in a circle and a third sphere intersect the circle at two places

Based on the model assumptions made, the vector B*i* that describes the elbow coordinates for each of the three arms as

$$B_i = [f + l_a\, cos(\theta_i) \quad 0 \quad l_a\, sin\, 1\, \theta_i i]^T \tag{2.12}$$

To calculate the direct kinematics, we move the center of the spheres to inside from points to the points for i=1,2 and 3 respectively. After this transition the three spheres will intersect in the TCP point.

$$B_i' = [(f - e) + l_a\, cos(\theta_i) \quad 0 \quad l_a sin\,(\theta_i) i]^T \tag{2.13}$$

Where $e = B_1 B_1' = B_2 B_2' = B_3 B_3' = 0$ is the length of shifted distance, clearly described in Fig.2.8 To achieve a matrix that describes all of the three points in the base frame {**O**} one has to multiply B_i with the rotational matrix R_z^0:

$$\boldsymbol{R_z^0} = \begin{bmatrix} cos(\theta_1) & sin(\theta_1) & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.14)}$$

The result is the matrix $\mathbf{B'}$,

$$\mathbf{B'} = \mathbf{R_z^0 B_i'} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} [(f-e) + l_a \cos(\theta_i) \quad 0 \quad l_a \sin(\theta_i)]^T$$

$$\mathbf{B'} = \begin{bmatrix} \cos(\alpha) B_{i,x}^{'} \\ \sin(\alpha) B_{i,x}^{'} \\ z' \end{bmatrix} = \begin{bmatrix} \cos(\alpha)[(f-e) + l_a \cos(\theta_i)] \\ \sin(\alpha)[(f-e) + l_a \cos(\theta_i)] \\ l_a \sin(\theta_i) \end{bmatrix} = \begin{bmatrix} S_{i,x} \\ S_{i,y} \\ S_{i,z} \end{bmatrix} \quad \textbf{(2.15)}$$

Then can three spheres be created with the forearm's lengths l_b as radius, and their centers in B_i respectively. The general equation for a sphere is

$$(x - S_{i,x})^2 + (y - S_{i,y})^2 + (z - S_{i,z})^2 = r^2 \quad \textbf{(2.16)}$$

This gives the three equations for three links *i* =1,2 and 3 respectively. For link (1) the upper arm is parallel to x- axis and perpendicular to y-axis, so the rotation angle α =0, but the other two links have a rotation angle α = 120 for link (2) and α = -120 for link (3).

$$(x - \cos(\alpha_1)[(f-e) + l_a \cos(\theta_1)])^2 + (y - \sin(\alpha_1)[(f-e) + l_a \cos(\theta_1)])^2 + (z - l_a \cos(\theta_1))^2 = l_b^2$$

$$(x - \cos(\alpha_2)[(f-e) + l_a \cos(\theta_2)])^2 + (y - \sin(\alpha_2)[(f-e) + l_a \cos(\theta_2)])^2 + (z - l_a \sin(\theta_2))^2 = l_b^2$$

$$(x - \cos(\alpha_3)[(f-e) + l_a \cos(\theta_3)])^2 + (y - \sin(\alpha_3)[(f-e) + l_a \cos(\theta_3)])^2 + (z - l_a \sin(\theta_3))^2 = l_b^2$$

(2.17)

After substitution the values α_1 =0, α_2= 120, and α_3=-120 in Eq. 13, we get the three sphere equations,

$$(x - [(f - e) + l_a \cos(\theta_1)])^2 + (y)^2 + (z - l_a \sin(\theta_1))^2 = l_b^2$$

$$(x + \frac{1}{2}[(f - e) + l_a \cos(\theta_2)])^2 + (y - \frac{\sqrt{3}}{2}[(f - e) + l_a \cos(\theta_2)])^2 + (z - l_a \sin(\theta_2))^2 = l_b^2$$

$$(x - \frac{1}{2}[(f - e) + l_a \cos(\theta_3)])^2 + (y - \frac{\sqrt{3}}{2}[(f - e) + l_a \cos(\theta_3)])^2 + (z - l_a \sin(\theta_3))^2 = l_b^2$$

(2.18)

Rearrange Eq. 2.18 we obtain,

$$(x + k_{11})^2 + (y + k_{12})^2 + (z + k_{11})^2 = l_b^2$$

$$(x + k_{21})^2 + (y + k_{22})^2 + (z + k_{23})^2 = l_b^2$$

$$(x + k_{31})^2 + (y + k_{32})^2 + (z + k_{33})^2 = l_b^2 \qquad \textbf{(2.19)}$$

Where,

$$k_{11} = (f - e) + l_a$$

$$k_{12} = 0$$

$$k_{13} = -l_a \sin(\theta_1)$$

$$k_{21} = \frac{1}{2}[(f - e) + l_a \cos(\theta_2)]$$

$$k_{22} = -\frac{\sqrt{3}}{2}[(f - e) + l_a \cos(\theta_2)]$$

$$k_{23} = -l_a \sin(\theta_2)$$

$$k_{31} = \frac{1}{2}[(f - e) + l_a \cos(\theta_3)]$$

$$k_{32} = -\frac{\sqrt{3}}{2}[(f-e) + l_a \cos(\theta_3)]$$

$$k_{33} = -l_a \sin(\theta_3)$$

After expanding Eq. 2.19 we obtain,

$$x^2 + y^2 + z^2 + 2k_{i1}\,x + 2k_{i2}\,y + 2k_{i3}\,z = l_b^2 - (k_{i1}^2 + k_{i2}^2 + k_{i3}^2)\ , i = 1,2,3 \qquad \textbf{(2.20)}$$

Subtract Eq.2.20 with $i = 2$ from Eq. 2.20 with $i = 1$, we obtain,

$2(k_{11} - k_{21})x+2(k_{12} - k_{22})y+2(k_{13} - k_{23})z= (k_{21}^2 + k_{22}^2 + k_{23}^2) - (k_{11}^2 + k_{12}^2 + k_{13}^2)$
(2.21)

Subtract Eq.2.20 with $i = 3$ from Eq. 2.20 with $i = 1$, we obtain,

$2(k_{11} - k_{31})x+2(k_{12} - k_{32})y+2(k_{13} - k_{33})z= (k_{31}^2 + k_{32}^2 + k_{33}^2) - (k_{11}^2 + k_{12}^2 + k_{13}^2)$
(2.22)

Simplifying Eq.2.21 and Eq.2.22 we obtain,

$$a_1\,x + b_1\,y + c_1\,z = d_1$$

$$a_2\,x + b_2\,y + c_2\,z = d_2$$

(2.23)

Where,

$$a_1 = 2(k_{11} - k_{21})$$

$$b_1 = 2(k_{12} - k_{22})$$

$$c_1 = 2(k_{13} - k_{23})$$

$$a_2 = 2(k_{11} - k_{31})$$

$$b_2 = 2(k_{12} - k_{32})$$

$$c_2 = 2(k_{13} - k_{33})$$

And,

$$d1 = (k_{21}^2 + k_{22}^2 + k_{23}^2) - (k_{11}^2 + k_{12}^2 + k_{13}^2)$$

$$d2 = (k_{31}^2 + k_{32}^2 + k_{33}^2) - (k_{11}^2 + k_{12}^2 + k_{13}^2)$$

Arranging Eq.2.23 in a matrix form, we obtain,

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} d_1 - c_1 z \\ d_2 - c_2 z \end{bmatrix} \quad \textbf{(2.24)}$$

Define $\Delta = a_1 b_2 - a_2 b_1$, then for case $\Delta \notin 0$,

$$\Delta_x = (d_1 - c_1 z) b_2 - (d_2 - c_2 z) b_1 = (b_2 d_1 - b_1 d_2) - (b_1 c_2 - b_2 c_1) z$$

$$\Delta_y = (d_2 - c_2 z) a_2 - (d_1 - c_1 z) a_1 = (a_1 d_2 - a_2 d_1) - (a_2 c_1 - a_1 c_2) z$$

$$x = \frac{\Delta_x}{\Delta} = \frac{b_2 d_1 - b_1 d_2}{\Delta} + \frac{b_1 c_2 - b_2 c_1}{\Delta} z$$

$$y = \frac{\Delta_y}{\Delta} = \frac{a_1 d_1 - a_2 d_1}{\Delta} + \frac{a_2 c_1 - a_1 c_2}{\Delta} z \quad \textbf{(2.25)}$$

Consider,

$$f_1 = \frac{b_2 d_1 - b_1 d_2}{\Delta}, f_2 = \frac{a_1 d_1 - a_2 d_1}{\Delta}$$

$$f_x = \frac{b_1 c_2 - b_2 c_1}{\Delta}, f_y = \frac{a_2 c_1 - a_1 c_2}{\Delta}$$

Then,

$$x = f_1 + f_x z$$

$$y = f_{21} + f_y z \quad \textbf{(2.26)}$$

Substituting Eq. 2.26 in Eq.2.19 for *i*=3; we obtain,

$$(1 + f_x^2 + f_y^2)z^2 + 2([f_x f_1 + f_x k_{31}] + [f_y f_2 + f_y k_{32}] + k_{33})z + f_{11}^2 + f_{22}^2 + k_{33}^2 - l_b^2 = 0 \qquad \mathbf{(2.27)}$$

Let,

$$A = (1 + f_x^2 + f_y^2)$$

$$B = 2([f_x f_1 + f_x k_{31}] + [f_y f_2 + f_y k_{32}] + k_{33})$$

$$C = f_{11}^2 + f_{22}^2 + k_{33}^2 - l_b^2$$

Where,

$$f_{11} = f_1 + k_{31}$$

$$f_{22} = f_2 + k_{32}$$

The solution of Equation $Az^2 + Bz + C = 0$ is well known as

$$Z = \frac{B \pm \sqrt{B^2 - 4AC}}{2A} \qquad \mathbf{(2.28)}$$

From Eq.2.28, we can Evaluates Eq.2.26.

Mathematically neither forward nor inverse kinematics gives single solution. Forward kinematics usually has two solutions, because the passive joint angles formed between upper arm and lower arm are not determined by kinematic equations. Then the solution that is within the robot's work area must be chosen. With the base frame {O} in this case, it will lead to the solution with negative z coordinate. The output solution has Four possible cases:

1) Generic solution. The two solutions are realized at the intersection of a circle and a sphere.
2) Singular solution. Once sphere is tangent to the circle of intersection of the other two spheres, hence there is only one solution possible.
3) Singular solution. The center of any two spheres coincides, resulting in an infinite number of solutions. This is an unlikely configuration for most practical embodiments of the manipulator, except for the situation when $\boldsymbol{\theta_1} = \boldsymbol{\theta_2} = \boldsymbol{\theta_3} = \pi/2$.
4) No solution. The three spheres do not intersect at a common point.

4. CTPack Plants

In This section, we will present the detailed description of CTPack plants where delta robots are used.

4.1 CTPack Company Lines

It is a company which is specialized in automation advanced technology for food Packaging.

4.2 Areas of Expertise

Packaging, flow pack, secondary packaging, Pick & Place System, Wrapping System, Feeding System, Ice Cream, Bakery, Chocolate e turn-key packaging system.

Packaging Categories:

- **Ice Cream;** as the market's leading developer for automatic robotic lines for the ice cream industry, CT Pack boasts the longest tradition and the most profound knowledge of end-of-line applications for the ice cream business. Today CT Pack is the reference company for medium and high capacity, state of the art equipment.
- **Chocolate & Confectionery**; Specialized in packaging lines for chocolate & confectionery as single-lane wrappers flow pack can handle more than 1,000 ppm per leg, while multilane wrapping systems can manage up to 5,000 ppm.
- **Bakery**; Considering complete packaging lines for the bakery from a simple horizontal single lane wrapping unit to a comprehensive, fully integrated high-capacity line. CT Pack has the engineering skills and experience to provide efficient and user-friendly solutions.
- **Frozen Food;** considering complete packaging lines, frozen food for Multi-flavor boxes, top-loading, and side-loading multipack and impulse cartons are erected and closed by high-speed hot melt cartoning machines, demonstrating the completeness of CT Pack's secondary packaging solutions.
- **Pouches and Bags**; customized machines for packaging pouches and bags are available for high-speed production lines. Robotic systems with vision technology and compact box loading units allow flexible case packing with products standing-up, flat, and interleaved in outer cases, display boxes, and tray/hood cartons.

CT Pack also considers Solutions for other applications. The auto stacker for LC/OC coolers is an entirely automatic system for assembling automotive coolers and preparing them for process treatment on pallets.

The robotic system, guided by vision technology, handles various components for all major car types, without any format change needed and with maximum precision.

CT Pack packaging machines can be separated into three systems, as shown in the figure below: Feeding Systems, Primary Packaging Systems, and Robotic Solution Systems.

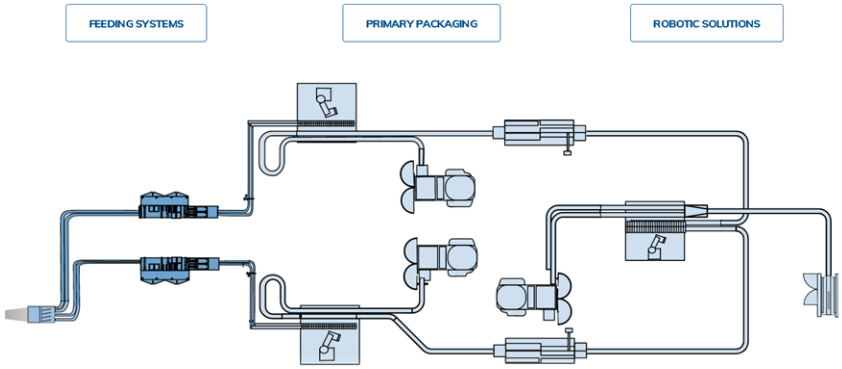


Figure 4.1 Planimetry of the production lane

Feeding System

The fully automatic system is primarily used in the confectionery industry (for ice cream, bakery, chocolate tablets and chocolate bars).

Primary Packaging

CT Pack designs and builds high speed flow-wrapping machines dedicated to different type of products, Fig. 3.5.

Figure 4.2 Flow Wrap machines single and dual lanes

Robotics Solutions:

Robotics Solutions have become a commonplace application in today's facilities, which deliver proven advantages for manufacturers, and relieve workers of monotonous, repetitive work.

The consistency at which a pick and place robot is able to complete assembly, quality control, packaging, and other material handling processes improves the overall quality of production and reduces downtime due to errors. Speed contributes significantly to productivity, as pick and place robots move products through the manufacturing process much quicker than manual options.

Pick and place robot grabs a part on an incoming conveyor belt and, rather than assemble the part, the robot places it in a packaging container at high speed.

4.3 Industrial Robotics Solutions

Case Packer Machine:

It is an Integrated system for forming, loading, and closing.

Figure 4.3 Case Packer Machine

The **Case Packer** consists of a single integrated unit with three sections:

- Box erector (forms the box and closes the bottom)
- Product-Handling, collating, top-loading **pick& place**.
- Box closing and sealing

This compact, fully integrated case packer is designed to meet the needs for low- to mid-capacity case packing, focusing on quality and footprint savings.
The Case is erected positively, assuring the empty carton's complete control through the whole case-packing process.

This machine's delta robot is responsible for picking the boxes from an operation conveyor station and placing them on a fixed station.

All in One Box Loading System:

High speed integrated box loading system. This modular machine forms and fills, and seal boxes from a flat blank. It integrates a box former, a collating unit with a top-loading robot, and a hot-melt box sealer.

Entirely suitable for a compact secondary packaging solution, the machine provides different features (double collating system, high-speed feeding)

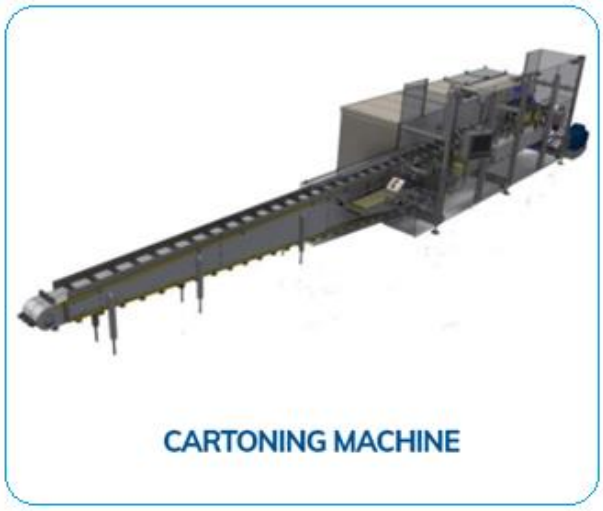


Figure 4.4 Cartoning Machine

This machine’s delta robot is responsible for picking the multiple products by a considerable Gripper from a fixed buffer station and placing them in multiple boxes on an operation station.

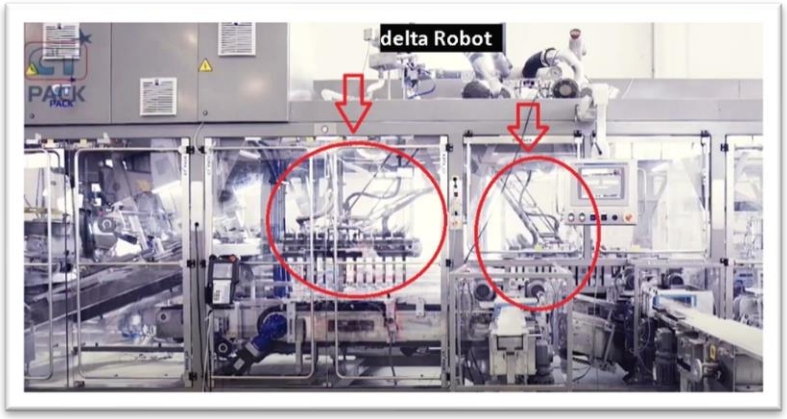


Figure 4.5 Position of multiple delta robot in the machine

Delta Robot CTFLEX

CTFLEX defines all the applications, which includes delta robots grouped in lines or clusters.
The application used for tray-box-case loading and the concept is modular to handle any product's type and production line speed.
The new robot is designed for more efficient use of smart multi-picking tools and will be equipped with a centralized control system to manage picking and placing operations and tray-box-case handling.
The delta robots have a modular composition and structure, allowing all possible vision systems for product recognition (photocells, IR systems, 3D vision).

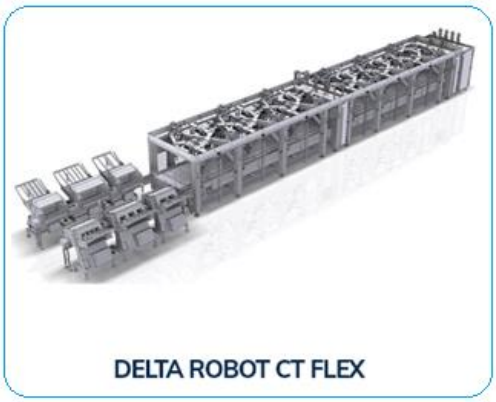


Figure 4.6 Multiple robot CTFLEX

Figure 4.7 Inside view of the multiple CTFLEX

Delta Robot “TLMP” Feeding System

The TLMP (Top-Loading Multi-Pack) system has been developed to feed, by top-loading operations, a Multipack wrapping machine or a horizontal cartoning machine (i.e., feed a continuous motion infeed chain or bucket conveyor). The system concept is modular, as it can be comprised of several robots, each one handling one incoming flow of products.

Figure 4.8 Delta Robot Unit TLMP

5. Multi-Robot Pick & Place Applications

This chapter aims to introduce all the background necessary to understand the principles behind the pick and place systems, where delta robots have fundamental roles. The essential type requested in pick and place solution systems is the Delta robot due to the flexibility of moving in a large area and its high interaction speed.

In recent years, the customers' demand of productivity and flexibility for their production lines has largely increased. This is why delta robots and robotics pick & place cells are more and more present in some industrial fields such as the food industry. In high-performance applications, typical characteristic of a pick & place robot can reach the following values: velocity 10 m/s, acceleration 100 m/s2 precision +/- 0.1mm, pick & place cycle 0.40s on average [16]. To improve the performance of these applications it is necessary to improve the design of current production systems (number of robots, performance ...) whilst also improving the management of flows and workload management when several robots are used. A pick & place application is usually composed of a series of several robots installed in a line one after the other taking products on a first conveyor and placing them in boxes located on a second conveyor, see figure 5.1, [15]. On a multi-robot packaging cell, when there is no workflow optimization system," pick" instructions are divided equally between the first robots. A final robot is added to deal with the products that could not be taken by the previous robots. Products initially assigned to a robot may not be taken because they finally are out of the robot workspace because of a lack of boxes to fill, for example.

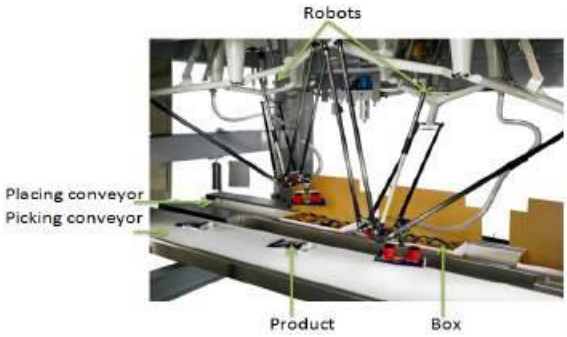


Figure 5.1

5.1 General structure of the pick-and place system

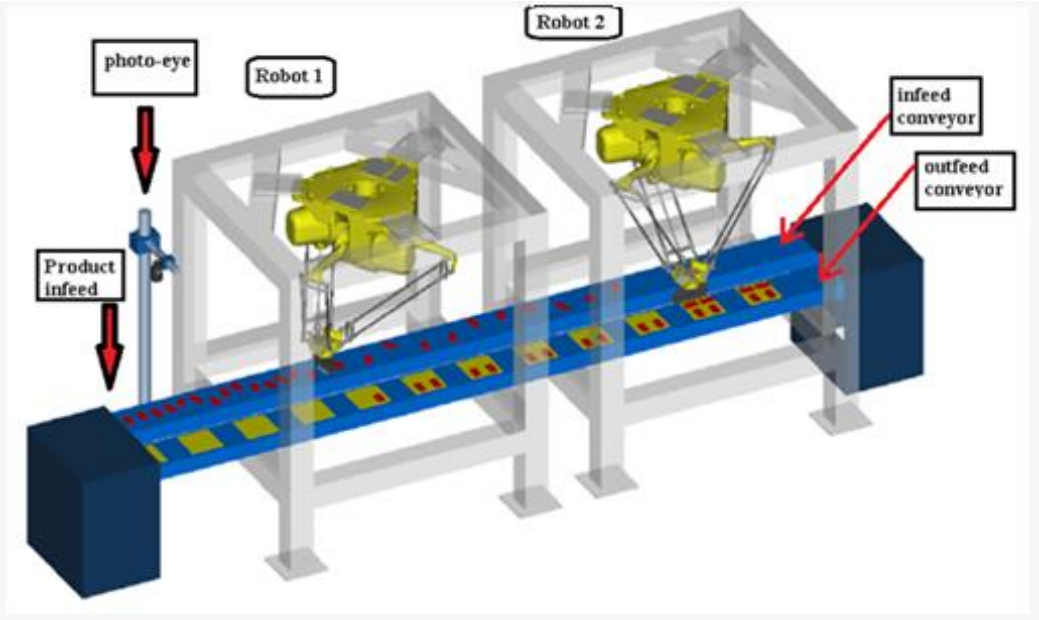


Figure 5.2 Structure of the machine

A robot operates as conveyor or fixed station. There are two types of operations: pick and place. There is one-to-one correspondence between a conveyor station and an operation or between a fixed station and an operation. When a conveyor or fixed station is generated, an operation is also generated automatically.

An encoder is a device that is used to provide feedback. That senses “position” of the Conveyor. Encoders will use motion under a variety of technologies, and translate it into an electrical signal. That signal is then sent back to a controlling device, such as a PLC, and is interpreted, meaning scaled, to represent a value that will then be used within the program

A sensor is an object which detects parts flowing on a conveyor and inputs the detection result to the most upstream conveyor station defined by the relevant conveyor object. It is available for detecting parts using a vision system and detecting them using a sensor such as a photo eye without using a vision system.

A conveyor is an ordered collection of the sensor and conveyor station objects (described later). A conveyor object indicates which sensor detects a part flowing on the conveyor and how it sequentially flows from a conveyor station to another conveyor station. A conveyor object virtually represents the actual conveyor status.

A conveyor station is an object indicating an area on a conveyor in which a robot operates on a moving part. The robot picks up the part from a conveyor station and places it in another conveyor station or fixed station. A conveyor station has a tracking frame and

tracking boundaries that limit the picking area. An example is given below, which will help understand conveyors, sensors, and conveyor stations more specifically. The following figure shows a visual tracking application consisting of two robots and two conveyors.

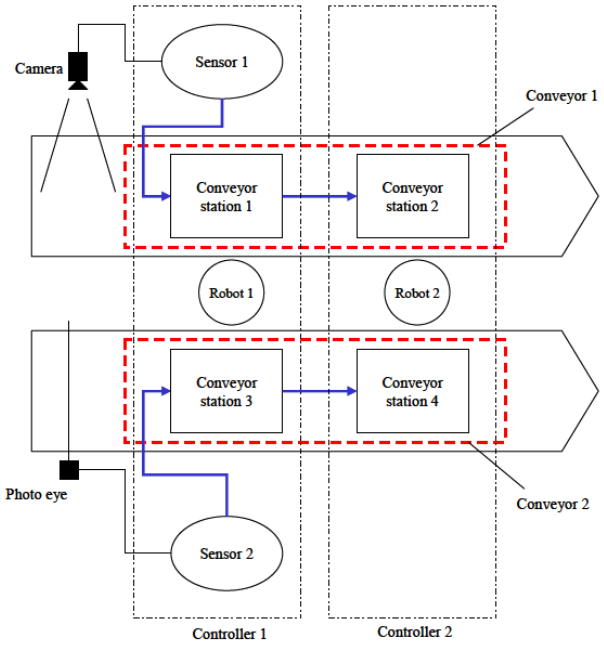


Figure 5.3 Conveyor station

Conveyor 1 is used as an infeed conveyor, and conveyor 2 is used as an outfeed conveyor. Both robots 1 and 2 pickup parts from conveyor 1 and place them on conveyor 2. One camera is installed at the upstream end of conveyor 1.

This camera is used to detect each flowing part. One photo-eye is installed at the upstream end of conveyor 2. This photo eye is used to detect each flowing box. This system picks up parts from conveyor 1 and places them in boxes flowing on conveyor 2.

This system has two conveyors, conveyors 1 and 2, four conveyor stations, and conveyor stations 1 to 4. Conveyor stations 1 and 2 are defined on conveyor 1 and conveyor stations 3 and 4 are defined on conveyor 2. Robot 1 picks up parts from conveyor station 1 on conveyor 1 and places them in conveyor station 3 on conveyor 2. Robot 2 picks up parts from conveyor station 2 on conveyor 1 and places them in conveyor station 4 on conveyor 2.

5.2 Multirobot coordination in pick-and place tasks on a moving conveyor

The first part of the system consists of the product infeed, where the products arrive on the infeed conveyor as separated products or grouped aligned products. Then, the photo-eye (Alternatively, camera vision) sensor detects these incoming products and assigns them to the robot's queue with the relatives of conveyor position. The sensor connected to the conveyor's crankshaft calculates the moving positions and sends this information in real-time to the controller of the robots. As soon as the moving part arrives at the conveyor station, the robot picks up the part from a conveyor station and places it in another conveyor station or fixed station, Fig. 5.2.

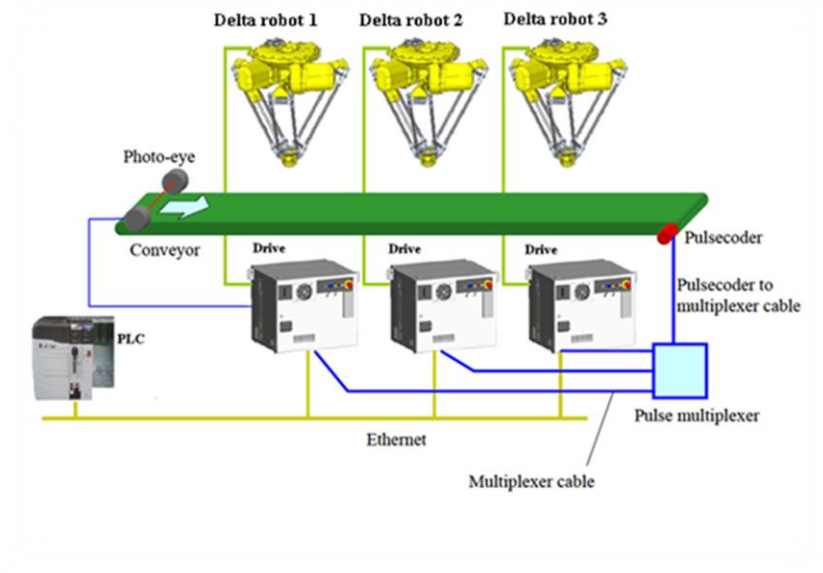


Figure 5.4 Tracking system

The robot software is programmed for applications that detect parts flowing on a conveyor using a camera or sensor such as a photo-eye and make robots pick up the parts as they move along the conveyor belt. This software provides functions for making multiple robots share the work of picking up parts flowing on one conveyor.
It also supports multiple conveyors, enabling a "conveyor-to-conveyor" application in which each robot picks up parts from one conveyor and places them on the other conveyor, and a "conveyor-to-fixed station" or "fixed station-to-conveyor" application in which not a conveyor, but a fixed station is used for infeed or outfeed. The robots can pick and place single or multiple parts from both these types of stations. The parts may be picked from or placed to trays. The robots can also pick from and place to buffers.

5.3 Part detection method

The first detection method uses the photo-eye, installed in the upstream area of the conveyor to detect a part by checking whether it passes in front of the photo-eye, Fig. 5.2. With a photo-eye, only the position in the conveyor moving direction can be recognized. The orientation and longitudinal conveyor position of flowing parts need to be constant

The second detection method uses the camera, installed in the upstream area of the conveyor, an image on the conveyor is snapped, and parts in the image are detected by the camera-Vision. A part placed at any position in any orientation on the conveyor can be picked up correctly because the position and orientation are recognized with the vision system.
When parts are detected using a camera, the following two methods are available: With one method, as the conveyor moves over a certain distance, the system snaps an image using the camera and continues monitoring the conveyor. With the other method, the system snaps an image using the camera to detect a part only when the part passes in front of a photo eye installed with the camera.

5.4 User Frame and User Tool

Position and posture of the robot are represented based on the frames. The user frame defines the working space for the robot to work. The user tool defines the position and orientation of the tooling (end effector). The origin of the user tool is also called TCP (Tool Center Point).
In robot systems that use a vision system, frames are very important. For instance, when the vision system returns the instruction to move 10 mm in X direction or to rotate 30 degrees around the Z-axis, the robot motion completely depends on the accurate definition of the frames.

The user frame defines the working space in which the robot works. The offset data from the vision system, (for example to move 10 mm in X direction or to rotate 30 degrees around the Z-axis,) are all respective to the user frame. Therefore, it is very important to teach the user frame as accurately as possible. If the user frame was set up inaccurately, the robot would move to an incorrect direction or rotate around an incorrect axis.
In the case of a 2-dimensional fixed-frame offset vision application, the user frame covers another important role. It defines the 2-dimensional work plane in the real 3-dimensional space. The 2-D work plane must be parallel to the X-Y plane of the user frame.

The user tool defines the position and orientation of the robot tooling (end effector). In a robot system that uses vision, it is very important to teach an accurate TCP (Tool Center Point) of the pointer tool that is used during teaching the user frame. If the TCP is less accurate, the taught user frame will also be less accurate.

When two or more robots work together, it is necessary to configure the system so that these robots physically share the same user frame. This is called the sharing of the user frame. Specifically, the sharing of the user frame is needed in the following cases:

- Multiple robots are offset with a single set of offset data.
- The robot to be offset is different from the robot that has the camera.

User frame sharing requires that all robots use the same user frame number. For example, user frame 5 of robot 1 needs to be physically the same as user frame 5 of robot 2.

5.5 Single Robot Delta

This chapter will present the delta robot's mechanical characteristics, its working area limitations, its cycle time, payload, and grippers that can carry. The figures show the in working Interface Area:

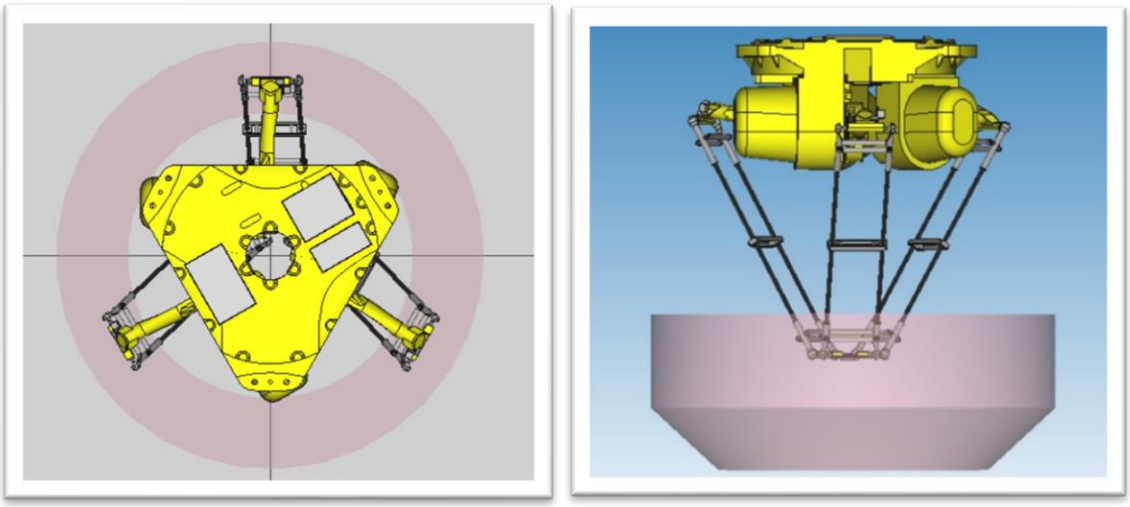


Figure 5.5 Delta Robot. Workspace

The movement of the robot in the working area can be limited by several factors such as design constraints of passive kinematic couples, and the limitations are given by the actuators, cohesive restrictions derived from the structural elements of the robot, as well as the points or areas of singularity, which may divide the working area of the various components, fig. 5.5.

5.6 Virtual Workplace Area

The best way to avoid the robot delta's physical limitation is to program a virtual workplace area between two virtual boundaries. The upper limit begins with the virtual upstream boundaries, and the lower Limits finish with the Downstream boundary.

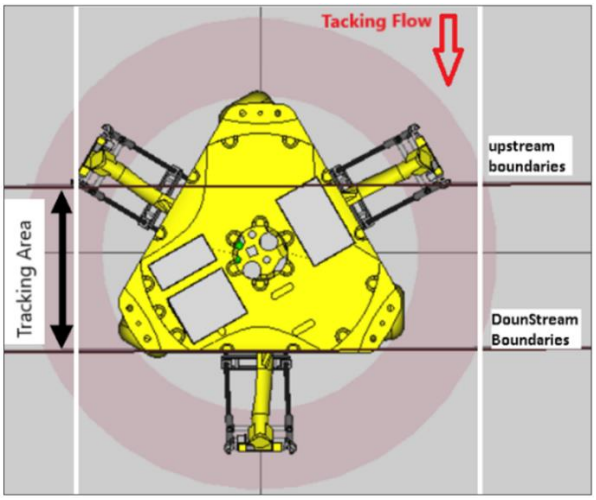


Figure 5.7 tacking area and boundaries

Tracking Area in the conveyor where the product flow is beginning from upstream and finishing with downstream; in this area, the robot pick/Place is in Tracking point, fig.5.7. And, the Discard Line Specify the position where the robot decides to discard a part carried on the conveyor. It is an offset from the downstream boundary of the tracking area, fig.5.8.

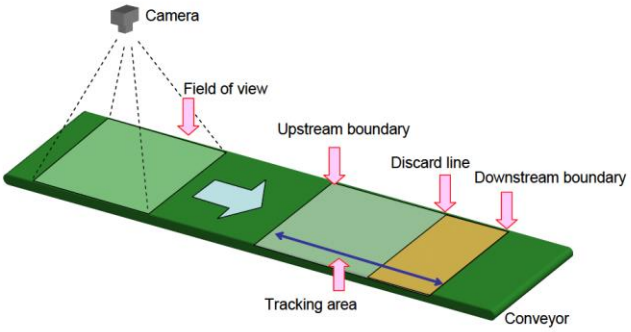


Figure 5.8 Boundaries and discard lines

5.7 Grippers

Grippers are active links between the equipment and the workpiece or in a more general sense between the grasping organs on a specific (normally the grippers fingers) and the object to be required. Their functions depend on a specific application and include:

- Temporary maintenance of a definite position and orientation of the workpiece relative to the gripper and the handling equipment.
- Retaining of static (wight), dynamic (Motion, acceleration or deceleration) or process specific forces and moments.
- Determination and change of position and orientation of the object to the handling equipment by means of wrist axes.
- Specific technical operations performed with, or in conjunction with, the gripper.

Gripper are subsystems of handling mechanism which provide temporary contact with the object to be grasped. They ensure the position and orientation when carrying and mating the object to the handling equipment. Prehension is achieved by force producing and form matching elements. The term “gripper” is also used in cases where no actual grasping, but rather holding of the object as e.g., in vacuum suction where the retention force can act on a point, line or surface.

Figure 5.9 Gripper

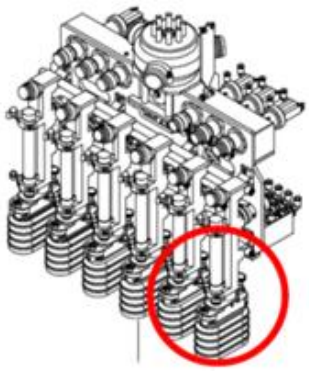


Figure 5.10 Gripper's parts

In pick-and place system, the delta robot usually has grippers with vacuum suction cups fig. 5.10, and these grippers' central function is in picking and placing the parts from/to conveyors.

5.8 Picking’s and Placing’s Steps:

Picking’s Steps:

- Products infeed on the track conveyor.
- eye sensor senses the product and put it inside the queue, every trigger has it own position related to track.
- the trigger moves on the track conveyor.
- before it arrives to the upstream boundary, the robot can get the trigger from the trigger from the queue.
- as soon as the trigger overcome the upstream boundary, the robot will follow the trigger in tracking (having the same speed of the track conveyor)
- if the trigger has been got before the Discard Line, then robot can continue to follow the trigger in order to pick it), taking in consideration the Downstream which is the maximum of the working Area of the Robots.
- If the trigger overcome the Discard Lines before it was got by the robot, the trigger will be lost, or skipped and restored in the queue in case multiple Robot, and the actual robot is not the last.

Placing:

After Picking the product by the robot, and in order to finish the cycle, it need to place the products picked in a fixed point or tracking point:

- If the placing is in fixed point, then as soon as the robot gets the okay signals from the controllers (means that its ready to be placed on).
- If the placing is a Tacking Trigger (in a tray or in a box), then it should follow the tracking order as in dynamic picking.

- And when the robot gets the queue and its positioning is above placing work space, then it will directly place on the tracking trigger.
- Otherwise, the robot couldn’t make in time and the trigger will be skipped.

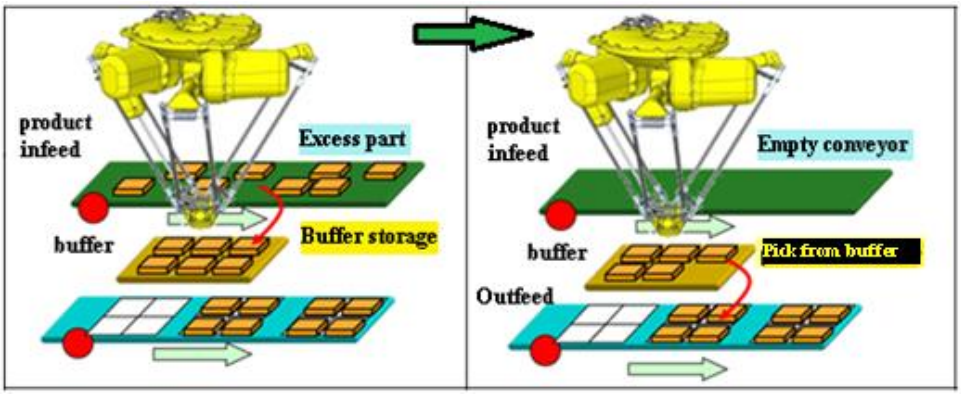


Figure 5.11 Delta's Robot Picking and Placing

NOTE:

If the space between the upstream boundary of the tracking area and the discard line is narrow, the situation is prone to occur where there is no part that can be allocated to the conveyor station. If there is no part that can be allocated to the conveyor station, the robot needs to wait until the next part passes the upstream boundary of the tracking area. If there is always at least one part between the upstream boundary of the tracking area and the discard line, the robot can pick up parts continuously. For example, if parts are supplied at almost even intervals, widening the space between the upstream boundary of the tracking area and the discard line makes it easier for the robot to pick up parts continuously, rather than supplying parts at even intervals.

6. Design and Implementation of CT Flex Delta Robot

6.1 Introduction

This chapter discusses the design and implementation of the new CT Flex robot, inspired from the traditional Delta robot. We focus mainly on the motivation behind development of this robot in comparison with the traditional Delta robot and its probable applications. Then we inspect the design aspects of this robot in details. We discuss its hardware design and derive the corresponding kinematic equations. After that, we propose our methodology for performing trajectory planning using this robot and introduce eventually the software platform used for programming it.

6.1.1 Motivation

CT Flex robot is designed for enhanced usage of multi-picking tools in comparison with traditional Delta robots, thanks to its intuitive design and cutting-edge grippers. It includes Delta robots grouped in lines or clusters and defines all the applications that are mainly used for tray/box/case loading. Furthermore, it is equipped with a centralized control system that manages picking and placing operations. The overall design is modular and can handle various product types and production-line speeds and can integrate numerous vision systems for product recognition. Figure 6.1 shows a CT Flex robot composed of 2 Delta robots.

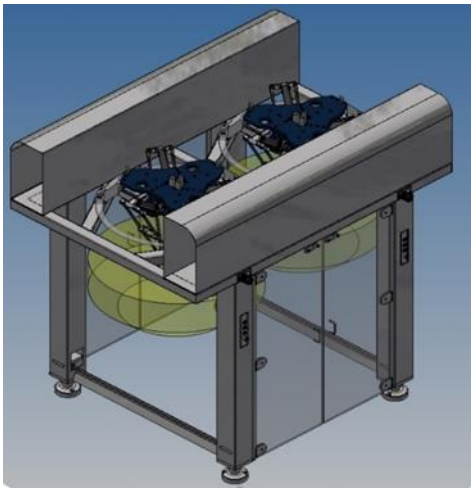


*Figure 6.*CT Flex robot composed of 2 Delta robots - 6.1.1.

Optimizing performance of pick and place tools while reducing the cost of building them is an essential target realized upon design of CT Flex robot. By cascading several Delta robots, a cost-effective system is developed using homogeneous hardware and software platforms, resulting in a

centralized control mechanism for the whole robot and thus eliminating the need to interface different robotic platforms.

6.1.2 Comparison with Traditional Delta Robot

The payload is the weight the robot can lift. The payload includes the weight of the End of Arm Tooling (EOAT) and the product weight to be picked. CT Flex robot can carry payloads up to 20kg and can reach a speed of 100 cycles per minute for 1 kg payloads in comparison to12 kg payloads and xxx cycles per minute for traditional Delta robots. Enhanced performance coupled with smoother operation are also noted for CT Flex robot in comparison with Delta robot. Moreover, it's essential to mention here that the materials used in building CT Flex robot are all lightweight, including usage of carbon fiber, anodized aluminum, stainless steel and plastics.

6.2 Hardware Design

Throughout this section, we will be discussing the hardware elements used in the implementation of the CT Flex robot. More specifically, we will be discussing the actuators, gears, electric drives and programmable logic controllers used in the robot.

6.2.1 Actuators / Motors

The main actuator type used in the CT Flex robot is the Kinetix VP low inertia motor VPL-B1153F with a frame size ranging from 063 to 165 mm. This motor is a brushless AC synchronous servo motor with remarkable high torque to size ratio and low rotor inertia. It is designed with a single cable technology for ease of interface. It has a rated output ranging from 0.19 to 5.55 kW and a speed reaching to 8000 rounds per minute. The typical applications for this motor are: packaging, converting, material handling, automotive and electronic assembly. Figure 6.2 shows a screenshot of the used Kinetix VP motor in CT Flex robot.

Figure 6.2 - Kinetix VP Motor used in CT Flex robot.

6.2.2 Gears

The gearbox used with the CT Flex robot is the TP+ Planetary gearbox widely implemented in traditional Delta robots. It is ideally suited for high positioning accuracy and highly dynamic cyclic operation. The torque provided with this gearbox is particularly well suited for high-precision applications in which high torsional rigidity is required.

At first, a gearbox ratio of 20 was used in the CT Flex robot, but provided limited performance capabilities and acceleration. A much better performance was obtained when we switched to use a gearbox ratio of 40. Figure 6.3 shows the TP+ gearbox used in our project.

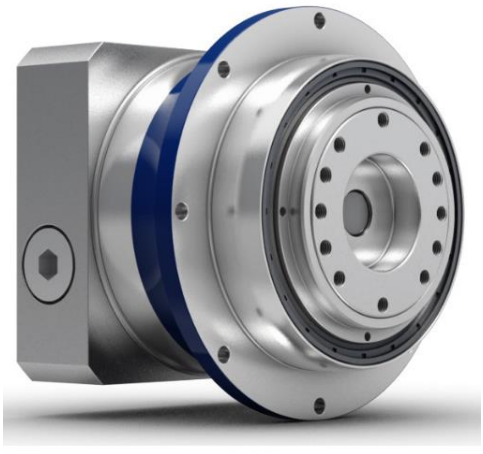


Figure 6.3 - TP+ gearbox used in CT Flex robot.

6.2.3 Electric Drives

The electric setup used to drive the Kinetix VP inertia motors of CT Flex robot is the Allen-Bradley 2198-D032-ERS3 dual axis inverter module (Figure 6.4). It is part of the Kinetix Ethernet I/P 5700 servo system. This drive delivers a continuous output power of 4.5 kW at 240V input, and 8.9 kW @ 480V input. It has continuous output current of 18.3 A and continuous peak current of 45.9 A. It supports maximum velocity loop of 400 Hz, Current loop of 1000 Hz and PWM frequency of 4 kHz.

Figure 6.4 - Electric Drive for actuators of CT Flex robot.

6.2.4 Programmable Logic Controllers

The programmable logic controller used in the CT Flex robot is the CompactLogix 5380 Controller. It provides a scalable control solution to address a wide variety of applications ranging from standalone systems to more complex systems with devices that are connected to the controller via an Ethernet network. The controller is mounted on a DIN rail and can monitor and control local and remote I/O modules. Figure 6.5 shows a snapshot of the programmable logic controller used in the CT Flex robot.

Figure 6.5 - CompactLogix 5380 programmable logic controller.

6.2.5 Mechanical Parts

Here we present the basic mechanical parts of the CT Flex robot. For each mechanical part, we provide its weight, length, center coordinates and moment of inertia. Figure 6.6 shows various mechanical parts used in the robot.

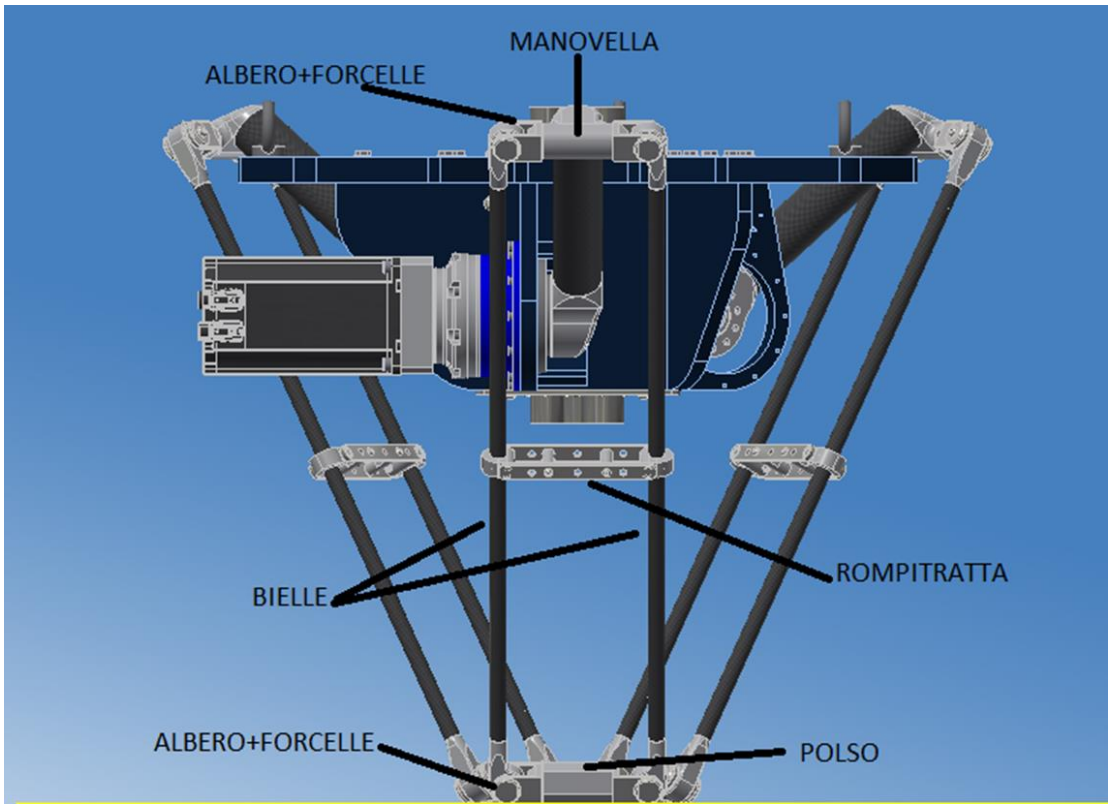


Figure 6.6 - Mechanical design of the CT Flex robot.

The basic mechanical components are as follows: crank (manovella), connecting rod (bielle), shaft and fork (albero and forcelle), breaker (rompiratta) and the wrist (polso) displayed in figures 6.7, 6.8, 6.9, and 6.10 respectively.

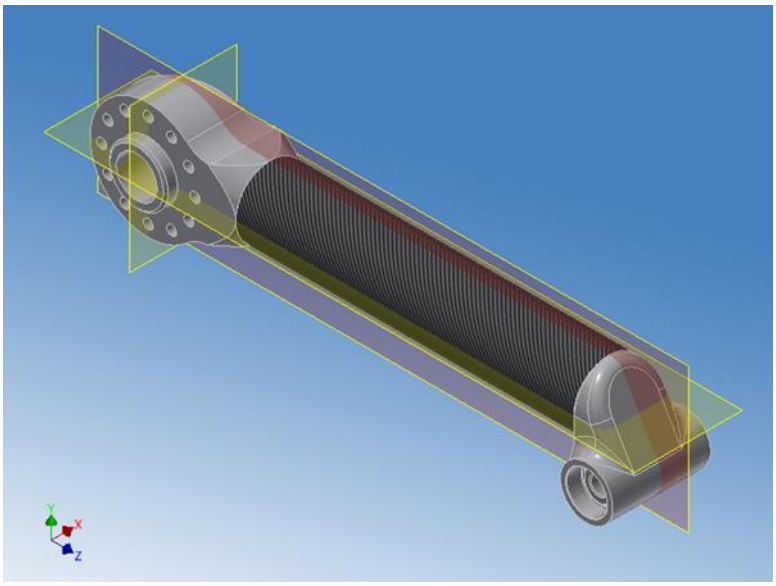


Figure 6.7 - Crank (manovella).

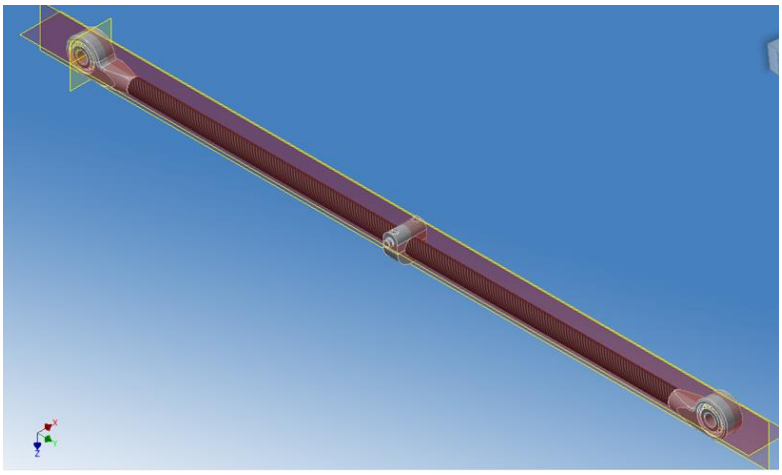


Figure 6.8 - Connecting rod (bielle).

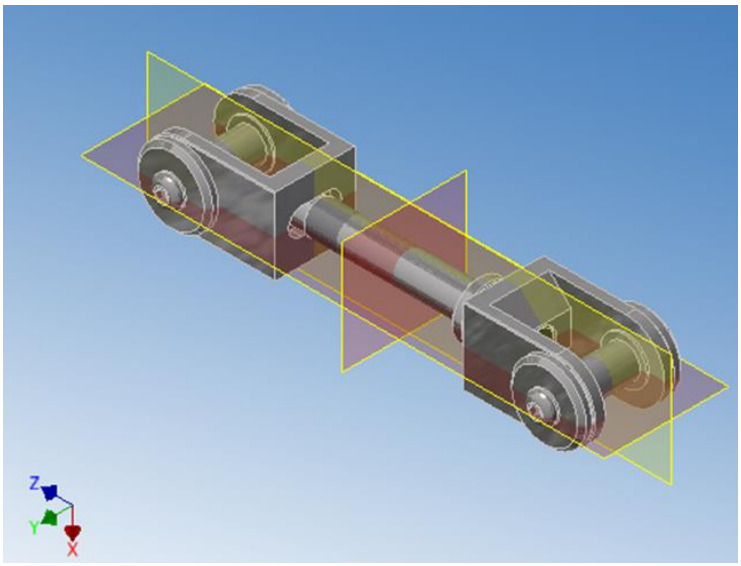


Figure 6.9 - Shaft and fork (albero and forcelle).

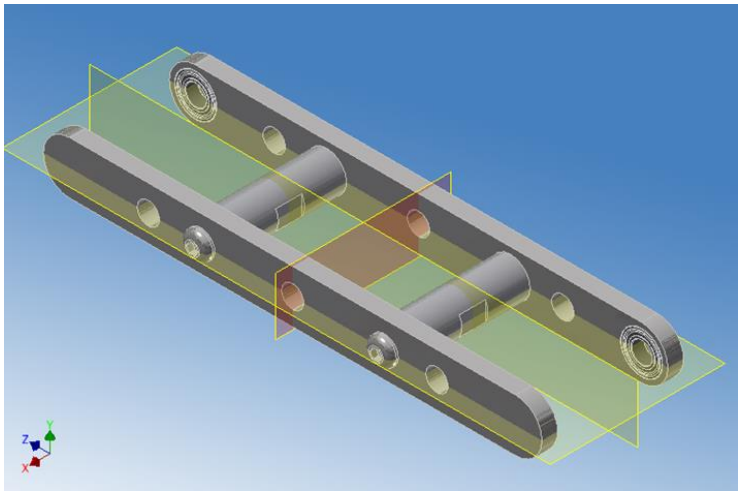


Figure 6.10 - Breaker (rompiratta).

Table 6.1 shows technical specifications of the used mechanical parts:

Table 6.1 - Technical information of mechanical components used in CT Flex robot.

Component	Mass	Length	Coordinates			Moment of Inertia					
	kg	mm	x	y	z	I_{xx}	I_{yy}	I_{zz}	I_{xy}	I_{xz}	I_{yz}
			mm	mm	mm	kg.mm^2	kg.mm^2	kg.mm^2	kg.mm^2	kg.mm^2	kg.mm^2
Crank	2.022	390	-3.23	-5.95	168.1	51769	51275	1815	43.842	-1187.5	2641.7
Connecting rod	0.681	870	0	435	5.69	86098	106.844	86063	-0.683	0	0
Shaft and fork	0.87	170	0	0.15	-0.52	4042.08	3937.06	224.792	0	0	-0.069

Breaker	0.355	170	0	0	0	830.135	1008.11	201.159	0	0	0
Wrist	1.561	90									

6.3 Kinematics of CT Flex Robot

Recall that the CT Flex robot is composed of clustered Delta robots. This means that the kinematic equations of the CT Flex manipulator are mainly those of the traditional Delta robot. Consequently, the approach for deriving the kinematic equations that govern the position and orientation of end-effector frame with respect to non-moving base frame in function of joint variables is basically the same as presented in chapter 3 of this thesis. These kinematic equations can be used to derive the forward kinematics whenever the joint variables values are provided, and can be further used to compute joint variables values whenever a target position and orientation of end-effector in workspace is desired (inverse kinematics). Figure 6.11 shows Delta robot diagram with the assignment of joint variables.

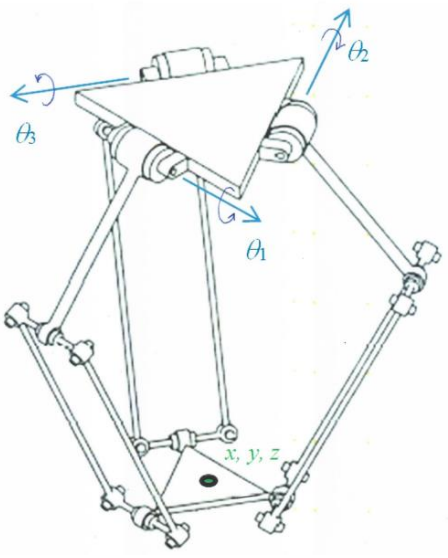


Figure 6.11 - Delta robot diagram.

6.4 Dynamics of CT Flex Robot

6.4.1 Assumptions

In this section, we discuss the equations of motion for the Delta robot. Before we proceed, we have to take into consideration the following assumptions:

- All links are rigid.

- All robot legs are identical.
- There exists no friction in the joints.
- The end effector stays perfectly level to the ground.
- The robot is mounted upside down (on the ceiling) and gravity acts downward.
- The two secondary links (forearms) are modelled as one link as they are constrained to move together.
- The mass of the joint at the elbow is modelled as a point mass.
- The second link (forearm) inertia is modelled as a uniform rigid slender bar.
- The center of mass of the first link (upper arm) and second link (forearm) is halfway along the link.
- There are no external forces or moments (other than gravity) acting on the end effector.

6.4.2 Lagrangian Dynamics

The Lagrangian, *L*, of a dynamical system is a function that summarizes the dynamics of the system. It is defined as:

$$L = T - U$$

Where *T* is the system's kinetic energy and *U* represents the system's potential energy. For a given inertial (fixed) coordinate system, the Lagrange's equation for unconstrained coordinate's states:

$$Q_j = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \frac{\partial L}{\partial q_j}$$

Where Q_i is a generalized force and q_i are the generalized coordinates, and *j* is a counter that runs from 1 to the number of degrees of freedom of the system (which in this case is the same as the number of generalized coordinates). For the particular case of the Delta robot (as shown in Figure 1 and 2) it is advantageous to use a constrained set of generalized coordinates to make finding the Lagrangian easier. For the Delta robot, the generalized coordinates are used where definition of the variables are given in the figures.

$$q = \{q_1, q_2, q_3, q_4, q_5, q_6\} = \{x, y, z, \alpha_1, \alpha_2, \alpha_3\}$$

The equivalent Lagrange's equation for constrained generalized coordinates is:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \frac{\partial L}{\partial q_j} = Q_j + \sum_{i=1}^{k} \lambda_i A_{ij}$$

Where:

λ_i = Lagrange multiplier

$A_{ij} = \frac{\partial f_i}{\partial q_j}$ with f_i being the constraints equations, so that $\sum_{i=1}^{k} \lambda_i A_{ij}$ is the generalized force of constraint.

k is the number of constraint equations.

Particularly for the Delta Robot, the constraint equation can be found by a vector loop on each leg which results in 3 constraint equations ($k = 3$). Performing a geometric analysis of the legs yields:

$$f_i = \left[\left(R + L_a \cos\alpha_i\right)\cos\vartheta_i - x\right]^2 + \left[\left(R + L_a \cos\alpha_i\right)\sin\vartheta_i - y\right]^2 + \left[\left(-L_a \sin\alpha_i\right) - z\right]^2 - {L_b}^2 = 0$$

Where:

$$R = R_a - R_b$$

This is also the equation of the sphere centred in the end effector (note that $R = R_a - R_b$) having L_b as ray, intersected by the three Delta motorized arms. This equation can also be utilized to solve for the forward and inverse kinematics.

6.4.3 Determination of Particular Terms

Substituting the generalized coordinates into Lagrange's constrained equation yields 6 equations:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = Q_1 + \sum_{i=1}^{k} \lambda_i A_{i1}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right) - \frac{\partial L}{\partial y} = Q_2 + \sum_{i=1}^{k} \lambda_i A_{i2}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{z}}\right) - \frac{\partial L}{\partial z} = Q_3 + \sum_{i=1}^{k} \lambda_i A_{i3}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_1}\right) - \frac{\partial L}{\partial \alpha_1} = Q_4 + \sum_{i=1}^{k} \lambda_i A_{i4}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_2}\right) - \frac{\partial L}{\partial \alpha_2} = Q_5 + \sum_{i=1}^{k} \lambda_i A_{i5}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_3}\right) - \frac{\partial L}{\partial \alpha_3} = Q_6 + \sum_{i=1}^{k} \lambda_i A_{i3}$$

We now need to determine each term given above and then differentiate them to find the solution.

6.4.4 Generalized Forces Q

The generalized forces for this can be determined by inspection. Due to the assumption that there are no external forces on the end effector:

$$Q_1 = Q_2 = Q_3 = 0$$

Due to the external torques on the joints:

$$Q_4 = \tau_1$$
$$Q_5 = \tau_2$$
$$Q_6 = \tau_3$$

6.4.5 Kinetic Energy

The kinetic energy, expressed in terms of the generalized coordinates, is composed of three terms:

$$T = \sum_{i=1}^{3} T_{La_i} + \sum_{i=1}^{3} T_{Lb_i} + T_{EE}$$

Where

$T_{EE} = \frac{1}{2} M_{EE} v_{EE}^2$ (Kinetic energy of the end effector)

$T_{Lb_i} = \frac{1}{2} M_{Lb} \left[\frac{1}{3} \left\| v_{b_i} - v_{c_i} \right\|^2 + \left(v_{b_i}^T v_{c_i} \right) \right]$ (Kinetic Energy of the lower arm.

$T_{La_i} = \frac{1}{2} J_{La} \dot{\alpha}_i^2 + \frac{1}{2} M_C L_a^2 \dot{\alpha}_i^2$ (Kinetic energy of upper arm)

And

J_{La} is the inertia of the upper arm (including motor inertia) *around the pivot point*.

M_C is the mass of the connecting joint between the upper arm and lower arm

M_{L_b} is the mass of the upper arm

M_{EE} is the mass of the end effector

$v_{EE} = (v_x \;\; v_y \;\; v_z)^T = (\dot{x} \;\; \dot{y} \;\; \dot{z})^T$ (Velocity of the end effector)

These now need to be expressed solely in terms of the generalized coordinates. The first and third terms are straightforward, and only the second term needs to be further explained. For this term, it is easier to relate the end effector velocity into the plane of the leg. Thus, in a rotated coordinate system

$$v_{bi} = \begin{Bmatrix} v_x \cos\vartheta_i - v_y \sin\vartheta_i \\ v_y \cos\vartheta_i + v_x \sin\vartheta_i \\ v_z \end{Bmatrix}$$

$$v_{ci} = \begin{Bmatrix} -\dot{\alpha}_i L_a \sin\alpha_i \\ 0 \\ -\dot{\alpha}_i L_a \cos\alpha_i \end{Bmatrix}$$

Therefore

$$\begin{aligned} T_{Lb_i} &= \frac{1}{2} M_{Lb} \left[\frac{1}{3} \left\| v_{bi} - v_{ci} \right\|^2 + \left(v_{b_i}^T v_{ci} \right) \right] \\ &= \frac{1}{2} M_{Lb} \left[\frac{1}{3} \left(v_x \cos\vartheta_i - v_y \sin\vartheta_i + \dot{\alpha}_i L_a \sin\alpha_i \right)^2 + \frac{1}{3} \left(v_y \cos\vartheta_i + v_x \sin\vartheta_i \right)^2 + \frac{1}{3} \left(v_z + \dot{\alpha}_i L_a \cos\alpha_i \right)^2 \right. \\ &\quad \left. + \left(- \left(v_x \cos\vartheta_i - v_y \sin\vartheta_i \right) \dot{\alpha}_i L_a \sin\alpha_i \right) - \left(v_z \dot{\alpha}_i L_a \cos\alpha_i \right) \right] \end{aligned}$$

And after simplification and putting in terms of the generalized coordinates:

$$\begin{aligned} T_{Lb-i} &= \frac{1}{6} M_{Lb} \left[v_x^2 + v_y^2 + v_z^2 + L_a^2 \dot{\alpha}_i^2 + \left(L_a \sin\alpha_i \left(-v_x \cos\vartheta_i + v_y \sin\vartheta \right) \dot{\alpha}_i \right) - \left(v_z \dot{\alpha}_i L_a \cos\alpha_i \right) \right] \\ &= \frac{1}{6} M_{Lb} \left[\dot{x}^2 + \dot{y}^2 + \dot{z}^2 + L_a^2 \dot{\alpha}_i^2 + \left(L_a \sin\alpha_i \left(-\dot{x} \cos\vartheta_i + \dot{y} \sin\vartheta \right) \dot{\alpha}_i \right) - \left(\dot{z} \dot{\alpha}_i L_a \cos\alpha_i \right) \right] \end{aligned}$$

6.4.6 Potential Energy

Similar to the kinetic energy, the potential energy is a sum of three terms:

$$U = U_{EE} + \sum_{i=1}^{3} \left(U_{ai} + U_{bi} \right)$$

All these terms can be further defined as:

$$\begin{aligned} U &= M_{EE} g z + \sum_{i=1}^{3} \left(-\frac{1}{2} M_{La} g \left(L_a \sin\alpha_i \right) - M_C g L_a \sin\alpha_i + \frac{1}{2} M_{Lb} g \left(z - L_a \sin\alpha_i \right) \right) \\ &= M_{EE} g z + \frac{1}{2} \sum_{i=1}^{3} \left(M_{Lb} g z - M_{Lb} g L_a \sin\alpha_i - 2 M_C g L_a \sin\alpha_i - M_{La} g \left(L_a \sin\alpha_i \right) \right) \\ &= \left(M_{EE} + \frac{3}{2} M_{Lb} \right) g z - \frac{1}{2} \sum_{i=1}^{3} \left(\left(M_{Lb} + 2 M_C + M_{La} \right) g L_a \sin\alpha_i \right) \end{aligned}$$

6.4.7 Lagrangian

The complete formulation for the Lagrangian is:

$$L = \frac{1}{2} M_{EE} \left[\dot{x}^2 + \dot{y}^2 + \dot{z}^2 \right] + \frac{1}{2} \sum_{i=1}^{3} \left[\left(J_{La} \dot{\alpha}_i^2 + M_C {L_a}^2 \dot{\alpha}_i^2 \right) \right] + \frac{1}{6} M_{Lb} \sum_{i=1}^{3} \left[\dot{x}^2 + \dot{y}^2 + \dot{z}^2 \right]$$
$$+ \frac{1}{6} M_{Lb} \sum_{i=1}^{3} \left[L_a^2 \dot{\alpha}_i^2 + \left(L_a \sin \alpha_i \left(-\dot{x} \cos \vartheta_i + \dot{y} \sin \vartheta_i \right) \dot{\alpha}_i \right) - \left(\dot{z} \dot{\alpha}_i L_a \cos \alpha_i \right) \right]$$
$$- \left[\left(M_{EE} + \frac{3}{2} M_{Lb} \right) gz - \frac{1}{2} \sum_{i=1}^{3} \left(\left(M_{Lb} + 2M_C + M_{La} \right) g L_a \sin \alpha_i \right) \right]$$

6.4.8 Calculation of the Derivatives

Lagrangian Derivatives

From the equations above, the required derivatives for the Lagrangian are:

$$\left(\frac{\partial L}{\partial \dot{x}} \right) = \left(M_{EE} + M_{Lb} \right) \dot{x} + \frac{1}{6} M_{Lb} \sum_{i=1}^{3} \left[\left(-L_a \sin \alpha_i \cos \vartheta_i \dot{\alpha}_i \right) \right]$$

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{x}} \right) = \left(M_{EE} + M_{Lb} \right) \ddot{x} - \frac{1}{6} M_{Lb} L_a \sum_{i=1}^{3} \left[\cos \vartheta_i \left(\cos \alpha_i \dot{\alpha}_i^2 + \sin \alpha_i \ddot{\alpha}_i \right) \right]$$

$$\left(\frac{\partial L}{\partial x} \right) = 0$$

Similarly:

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{y}} \right) = \left(M_{EE} + M_{Lb} \right) \ddot{y} + \frac{1}{6} M_{Lb} L_a \sum_{i=1}^{3} \left[\sin \vartheta_i \left(\cos \alpha_i \dot{\alpha}_i^2 + \sin \alpha_i \ddot{\alpha}_i \right) \right]$$

$$\left(\frac{\partial L}{\partial y} \right) = 0$$

And

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{z}} \right) = \left(M_{EE} + M_{Lb} \right) \ddot{z} - \frac{1}{6} M_{Lb} L_a \sum_{i=1}^{3} \left[\left(\cos \alpha_i \ddot{\alpha}_i - \sin \alpha_i \dot{\alpha}_i^2 \right) \right]$$

$$\left(\frac{\partial L}{\partial z} \right) = - \left(M_{EE} + \frac{3}{2} M_{Lb} \right) g$$

For the final equations a recursion can be made with i =1...3. Thus:

$$\left(\frac{\partial L}{\partial \dot{\alpha}_i} \right) = \left(J_{La} \dot{\alpha}_i + M_C {L_a}^2 \dot{\alpha}_i \right) + \frac{1}{6} M_{Lb} \left[2L_a^2 \dot{\alpha}_i + \left(L_a \sin \alpha_i \left(-\dot{x} \cos \vartheta_i + \dot{y} \sin \vartheta_i \right) \right) - \left(\dot{z} L_a \cos \alpha_i \right) \right]$$
$$= \left(J_{La} + M_C {L_a}^2 + \frac{1}{3} M_{Lb} L_a^2 \right) \dot{\alpha}_i + \frac{1}{6} M_{Lb} L_a \left[\left(\sin \alpha_i \left(-\dot{x} \cos \vartheta_i + \dot{y} \sin \vartheta_i \right) \right) - \left(\dot{z} \cos \alpha_i \right) \right]$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_i}\right)=\left(J_{La}+M_C{L_a}^2+\frac{1}{3}M_{Lb}L_a^2\right)\ddot{\alpha}_i$$
$$+\frac{1}{6}M_{Lb}L_a\left[\left(\sin\alpha_i\left(\ddot{x}\cos\vartheta_i-\ddot{y}\sin\vartheta_i+\dot{z}\dot{\alpha}_i\right)\right)+\cos\alpha_i\left(\dot{x}\dot{\alpha}_i Cos\vartheta_i-\dot{y}\dot{\alpha}_i Sin\vartheta_i-\ddot{z}\right)\right]$$

And

$$\left(\frac{\partial L}{\partial \alpha_i}\right)=\frac{1}{6}M_{Lb}L_a\dot{\alpha}_i\left[\left(\cos\alpha_i\left(\dot{x}\cos\vartheta_i-\dot{y}\sin\vartheta_i\right)\right)+\left(\dot{z}\sin\alpha_i\right)\right]+\frac{1}{2}gL_a\cos\alpha_i\left(M_{Lb}+2M_C+M_{La}\right)$$

Constraint Derivatives

The constraint equations must also be differentiated to find the "*A*" coefficients. From the constraint equation, *f*, the derivatives are:

$$\frac{\partial f_i}{\partial x}=2\left[\left(R+L_a\cos\alpha_i\right)\cos\vartheta_i-x\right]\left[-1\right]=2\left[x-\left(R+L_a\cos\alpha_i\right)\cos\vartheta_i\right]$$

$$\frac{\partial f_i}{\partial y}=2\left[\left(R+L_a\cos\alpha_i\right)\sin\vartheta_i-y\right]\left[-1\right]=2\left[y-\left(R+L_a\cos\alpha_i\right)\sin\vartheta_i\right]$$

$$\frac{\partial f_i}{\partial z}=2\left[\left(-L_a\sin\alpha_i\right)-z\right]\left[-1\right]=2\left[z+L_a\sin\alpha_i\right]$$

$$\frac{\partial f_i}{\partial \alpha_i}=2\left[\left(R+L_a\cos\alpha_i\right)\cos\vartheta_i-x\right]\left(-L_a\sin\alpha_i\cos\vartheta_i\right)+2\left[\left(R+L_a\cos\alpha_i\right)\sin\vartheta_i-y\right]\left(-L_a\sin\alpha_i\sin\vartheta_i\right)$$
$$+2\left[\left(-L_a\sin\alpha_i\right)-z\right]\left(-L_a\cos\alpha_i\right)$$
$$=2L_a\left[\sin\alpha_i\left(x\cos\vartheta_i+y\sin\vartheta_i\right)+z\cos\alpha_i-R\sin\alpha_i\right]$$

One should note that due to the special nature of the constraint equations:

$$\frac{\partial f_i}{\partial \alpha_j}=0\quad i\neq j$$

Dynamic Equations of Motion

From the terms already determined, the final equations simplify to:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)=\sum_{i=1}^{3}\lambda_i A_{i1}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right)=\sum_{i=1}^{3}\lambda_i A_{i2}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{z}}\right)-\frac{\partial L}{\partial z}=\sum_{i=1}^{3}\lambda_i A_{i3}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_1}\right)-\frac{\partial L}{\partial \alpha_1}=Q_4+\lambda_1 A_{14}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_2}\right)-\frac{\partial L}{\partial \alpha_2}=Q_5+\lambda_2 A_{25}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_3}\right)-\frac{\partial L}{\partial \alpha_3}=Q_6+\lambda_3 A_{36}$$

Where all derivatives and constants are calculated above.

Inverse Dynamics

For the inverse dynamics, the position, velocity, and acceleration of the end effector is given and the goal is to determine the joint torques required for that motion. This is an algebraic problem. First, calculate the Lagrange multipliers via:

$$\underbrace{\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{31} \end{bmatrix}}_{A}\begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\end{bmatrix}=\underbrace{\begin{bmatrix} \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right) \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{z}}\right)-\left(\frac{\partial L}{\partial z}\right) \end{bmatrix}}_{b} \Rightarrow \begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\end{bmatrix}=A^{-1}b$$

And then by substitution into the final three equations:

$$\tau_1=Q_4=\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_1}\right)-\frac{\partial L}{\partial \alpha_1}-\lambda_1 A_{14}$$

$$\tau_2=Q_5=\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_2}\right)-\frac{\partial L}{\partial \alpha_2}-\lambda_2 A_{25}$$

$$\tau_3=Q_6=\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}_3}\right)-\frac{\partial L}{\partial \alpha_3}-\lambda_3 A_{36}$$

A potential problem occurs when trying to determine a *consistent* set of generalized coordinates. In other words, the constraint equation must not be violated. Thus, the position, velocity, and acceleration kinematics must be used to determine the set of "*q*'s" which are valid.

Forward Dynamics

For the forward dynamics, the joint torques are given functions of time and the position, velocity, and acceleration of the end effector is found. This turns out to be a set of second order nonlinear differential algebraic equations (DAE). Calculation of these can be done in a variety of ways. A straightforward manner is to take the equations above and reformulate them in the following form:

$$[M(q_j,t)]\{\ddot{q}\} = [A(q_j,t)]^T \{\lambda\} + \{F(q_j,\dot{q}_i,t)\}$$

Where curly braces are added to denote vector quantities and braces denote matrices. By differentiating the constraint equation twice to find an acceleration equation of the form:

$$[A]\{\ddot{q}\} + \left[\frac{dA}{dt}\right]\{\dot{q}\} = 0$$

The DAE can be solved by placing into a standard form first order ordinary differential equation (ODE) form:

$$\begin{bmatrix} [U] & [0] & [0] \\ [0] & [M] & -[A]^T \\ [0] & -[A] & [0] \end{bmatrix} \frac{d}{dt} \begin{Bmatrix} \{q\} \\ \{\dot{q}\} \\ \{\mu\} \end{Bmatrix} = \begin{Bmatrix} \{\dot{q}\} \\ \{F\} \\ \left[\frac{dA}{dt}\right]\{\dot{q}\} \end{Bmatrix}$$

Where

$$\mu_i = \int \lambda_i dt \Leftrightarrow \lambda_i = \dot{\mu}_i$$

This form can be solved using any standard first order ODE solver (e.g., ode45 in Matlab). The problem with this method is that it requires the acceleration coefficients of the constraint equation. For the Delta robot, they can be calculated from the velocity coefficients. This can be done by differentiating each term of the *A* matrix individually. Thus, for first element:

$$A_{11} = 2(x - (R + L_a \cos\alpha_1)\cos\vartheta_1)$$
$$\Rightarrow \frac{dA_{11}}{dt} = 2\dot{x} + 2L_a \sin\alpha_1 \cos\vartheta_1 \dot{\alpha}_1$$

Following a similar procedure for the rest of the first constraint:

$$\frac{dA_{12}}{dt} = 2\dot{y} + 2L_a \sin\alpha_1 \sin\vartheta_1 \dot{\alpha}_1$$
$$\frac{dA_{13}}{dt} = 2\dot{z} + 2L_a \cos\alpha_1 \dot{\alpha}_1$$
$$\frac{dA_{14}}{dt} = 2L_a \sin\alpha_1 \cos\vartheta_1 \dot{x} + 2L_a \sin\alpha_1 \sin\vartheta_1 \dot{y} + 2L_a \cos\alpha_1 \dot{z}$$
$$+ 2L_a(\cos\alpha_1(x\cos\vartheta_1 + y\sin\vartheta_1) - z\sin\alpha_1 - R\cos\alpha_1)\dot{\alpha}_1$$
$$\frac{dA_{15}}{dt} = 0$$
$$\frac{dA_{16}}{dt} = 0$$

Similarly for the second constraint:

$$\frac{dA_{21}}{dt} = 2\dot{x} + 2L_a \sin\alpha_2 \cos\vartheta_2 \dot{\alpha}_2$$

$$\frac{dA_{22}}{dt} = 2\dot{y} + 2L_a \sin\alpha_2 \sin\vartheta_2 \dot{\alpha}_2$$

$$\frac{dA_{23}}{dt} = 2\dot{z} + 2L_a \cos\alpha_2 \dot{\alpha}_2$$

$$\frac{dA_{24}}{dt} = 0$$

$$\frac{dA_{25}}{dt} = 2L_a \sin\alpha_2 \cos\vartheta_2 \dot{x} + 2L_a \sin\alpha_2 \sin\vartheta_2 \dot{y} + 2L_a \cos\alpha_2 \dot{z}$$
$$+ 2L_a(\cos\alpha_2(x\cos\vartheta_2 + y\sin\vartheta_2) - z\sin\alpha_2 - R\cos\alpha_2)\dot{\alpha}_2$$

$$\frac{dA_{26}}{dt} = 0$$

And for the third constraint:

$$\frac{dA_{31}}{dt} = 2\dot{x} + 2L_a \sin\alpha_3 \cos\vartheta_3 \dot{\alpha}_3$$

$$\frac{dA_{32}}{dt} = 2\dot{y} + 2L_a \sin\alpha_3 \sin\vartheta_3 \dot{\alpha}_3$$

$$\frac{dA_{33}}{dt} = 2\dot{z} + 2L_a \cos\alpha_3 \dot{\alpha}_3$$

$$\frac{dA_{34}}{dt} = 0$$

$$\frac{dA_{35}}{dt} = 0$$

$$\frac{dA_{36}}{dt} = 2L_a \sin\alpha_3 \cos\vartheta_3 \dot{x} + 2L_a \sin\alpha_3 \sin\vartheta_3 \dot{y} + 2L_a \cos\alpha_3 \dot{z}$$
$$+ 2L_a(\cos\alpha_3(x\cos\vartheta_3 + y\sin\vartheta_3) - z\sin\alpha_3 - R\cos\alpha_3)\dot{\alpha}_3$$

This can be more effectively coded using a loop, but is left in this form for explanatory purposes.

6.5 Software Configuration of CT Flex Robot

6.5.1 Software Used

The software used to configure the CT Flex robot is Studio 5000 Logix Designer. It is based on the Rockwell Automation Integrated Architecture that provides a common platform for programming of various controllers, including Allen-Bradley ControlLogix and CompactLogix. In addition, the software encompasses standard safety configurations along with the support of wide range of

communication, motion and input-output modules. The main features of its programming language are:

- Comprehensive instruction set.
- Symbolic programming.
- Structures and arrays.
- Ladder logic.
- Function block diagram.
- Sequential function chart editors.
- S88 equipment phase state model.

6.5.2 Axes Definition and Configuration

To start working with the CT Flex robot, its geometry has first to be configured using the Logix Designer software. The first step for doing so is by defining the axes of the robot in motion groups. Figure 6.12 shows a 4-axes Delta robot that moves in three-dimensional Cartesian space (X_1, X_2, X_3). Note that the number of joints greater than the degrees of freedom and that not all joints are actuated. These unactuated joints are cylindrical as projected in the CT Flex robot.

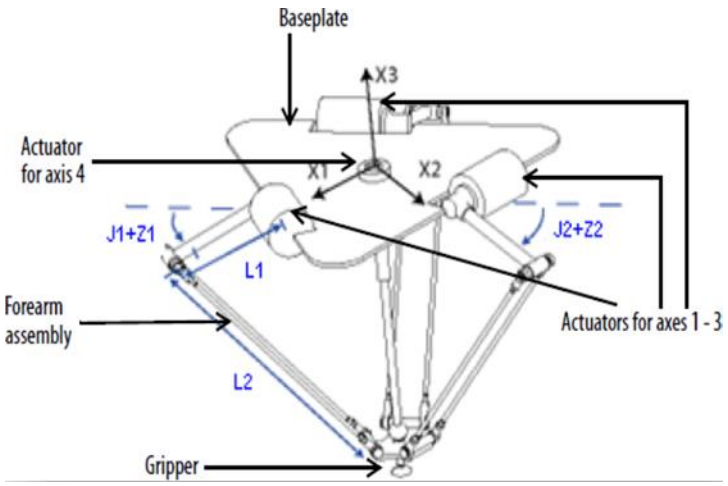


Figure 6.12 - 4-Axes Delta robot.

The Delta robot shown in the above figure has three degrees of freedom with an optional fourth degree of freedom used to rotate a part at the tooltip. In the Logix Designer application, the first three degrees of freedom are configured as three joint axes (J1, J2, J3) in the robot's coordinate system. The three joint axes are directly programmed in joint space and automatically controlled by the embedded kinematics software in Logix Designer.

This robot contains a fixed top plate and a moving bottom plate. The fixed top plate is attached to the moving bottom plate by three link-arm assemblies. All three link-arm assemblies have a single top link arm (L1) and a parallelogram two-bar link assembly (L2). As each axis (J1, J2, J3) is rotated, the center point of the gripper moves correspondingly in (X1, X2, X3) direction. The gripper remains vertical along the X3 axis while its position is translated to (X1, X2, X3) space by the mechanical action of the parallelograms in each of the forearm assemblies. The mechanical connections of the parallelograms via spherical joints ensure that the top and bottom plates remain parallel to each other.

Whenever you program the tool center point to a designated position (X1, X2, X3), the Logix Designer software computes the commands necessary for each of the joints (J1, J2, J3) to move the gripper linearly from the current position to the programmed (X1, X2, X3) position, at the programmed vector dynamics. When each top link (L1) moves downward, its corresponding joint axis (J1, J2, or J3) is assumed to be rotating in the positive direction. The three joint axes of the robot are configured as linear axes. Figure 6.13 shows the configuration of axes for Delta robot in motion groups.

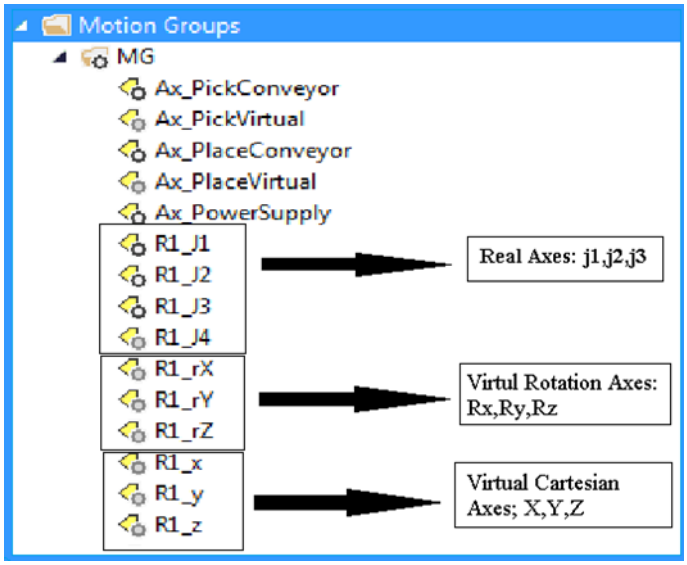


Figure 6.13 - Axes configuration in motion groups.

6.5.3 Establishing the Reference Frame

The reference frame for the Delta robot geometry is located at the center of the top fixed plate. Joint 1, Joint 2, and Joint 3 are actuated joints. If the Delta coordinate system in the Logix Designer software is configured with the joints home position at 0 in the horizontal position, then L1 of one of the link pairs will be aligned along the X1 positive axis. Moving in the counter-clockwise direction

from Joint 1 to Joint 2, the X2 axis will be orthogonal to the X1 axis. Based on the right-hand rule, X3 will be the axis pointing out of the paper in the positive direction (Figure 6.14).

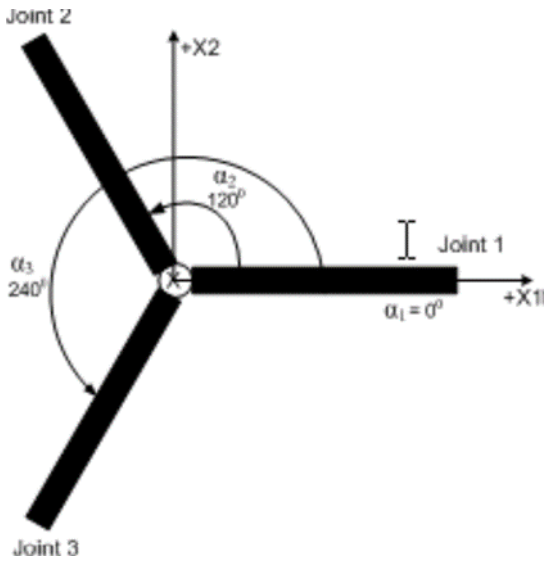


Figure 6.14 - Reference axis of Delta robot.

Figure 6.15 shows a snapshot from the software after configuration of joints and Cartesian coordinate systems.

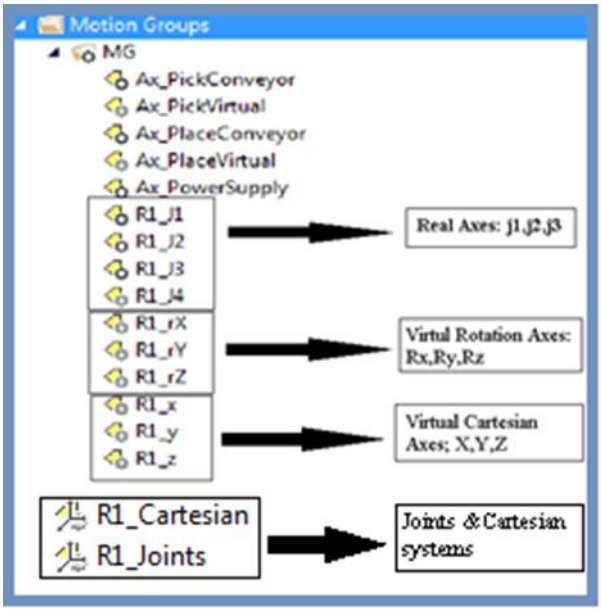


Figure 6.15 - Configuration of two coordinate systems in Logix Designer.

6.5.4 Configuring the Home Position

To configure the home position of the CT Flex robot, the following steps should be followed:

1. First, we need to obtain from the robot manufacturer, the angle values for J1, J2, and J3 at which the calibration position should be done. We will use these values to establish the reference position.
2. Next, we move all joints to the designated calibration position by jogging the robot under programmed control or manually by moving the robot when the joint axes are in an open-loop state.
3. After that, we set the configuration values for the joint axes home position to the calibration values obtained in step 1 and execute a Motion Axis Home (MAH) instruction for each joint axis.
4. Finally, we move each joint to an absolute position of 0.0 and then verify that each joint position reads 0 degrees and the respective L1 is in a horizontal position.

6.5.5 Workspace

Determining the workspace of the robot is a critical step prior to using it in the real world application. We start by setting the dimensions of top link arm (L1 = 390mm), the parallelogram two-bar assembly (L2 = 870mm), the base and effector plate (Figure 6.16).

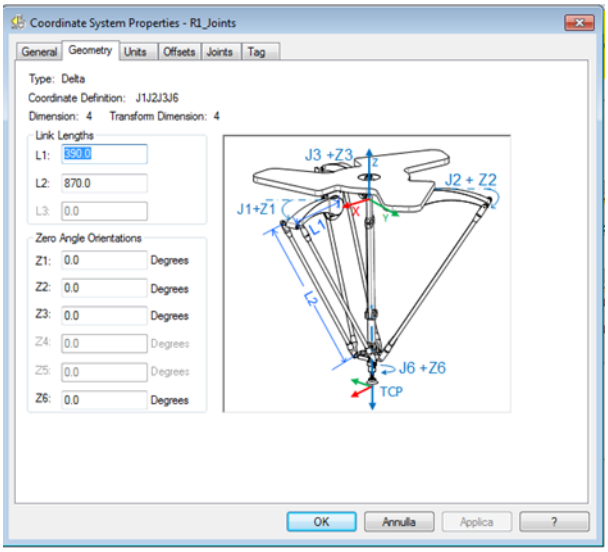


Figure 6.16 - Setting robot dimensions using Logix Designer.

The workspace of the robot is the volume comprising the coordinates of all points that the end-effector can reach. This workspace is limited because the physical robot's finite reach can collide between different parts of the robot as it moves. Using robot geometry defined earlier in the Logix Designer

software and the embedded kinematic equations (inverse), we proceed to calculate the robot workspace.

The workspace for an RRR (three rotational joints) planar robot where joint variable θ_1 is ranging between -40 and 90 degrees, with no limitation on values of joint variables θ_2 and θ_3. If there is no limit on θ_1, the workspace would be a hollow circle-shaped as θ_1 goes from 0 to 360 degrees. Similarly, if the links have the same dimensions, the workspace would be circular-shaped.

At the software level, we first define the initial reference system as described before, which has a coordinate point of (0, 0, 0) in the center of the upper plate. It is then translated into the tool center point of the end-effector. Recall that the robot tool center point (TCP) is used to create necessary adjustments that allow the controller to shift the coordinate system to keep track of the tool instead of the arm's end. An inaccurate center point will cause the robot to follow a different path than the one originally programmed.

Using the physical parts dimensions defined in the robot configuration, we calculate the following parameters: link lengths (L1, L2), dimensions of the base plate (Rb), dimensions of the end effector plate (Re) and the swing arm offset (D3).

Next, we need to calculate the upper and lower limits of the Z axis. In order to calculate the upper limit of Z axis in the Cartesian coordinate system, we apply the minimum value of the joints θ_1, θ_2, and θ_3= -40° (the minimum angle of the joints) to the Forward Kinematics equations. Figure 6.17 shows the Delta robot tool position at its highest point.

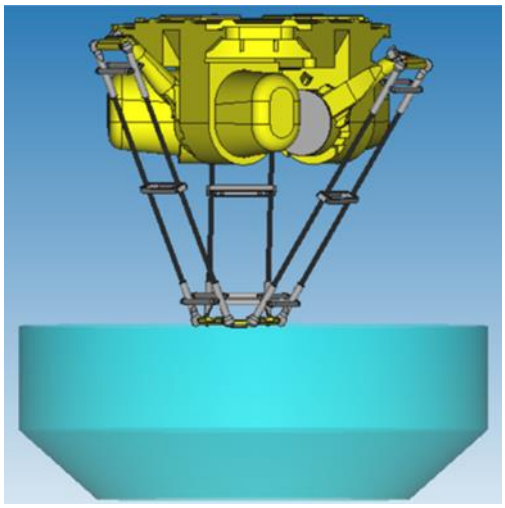


Figure 6.17 - Tool of Delta robot at its highest position.

In a similar fashion, we calculate the lower limit of the Z axis by applying the maximum value of the joint angle positions θ_1, θ_2, and θ_3= 90° in the forward kinematic equations. Figure 6.18 shows the Delta robot tool position at its lowest point.

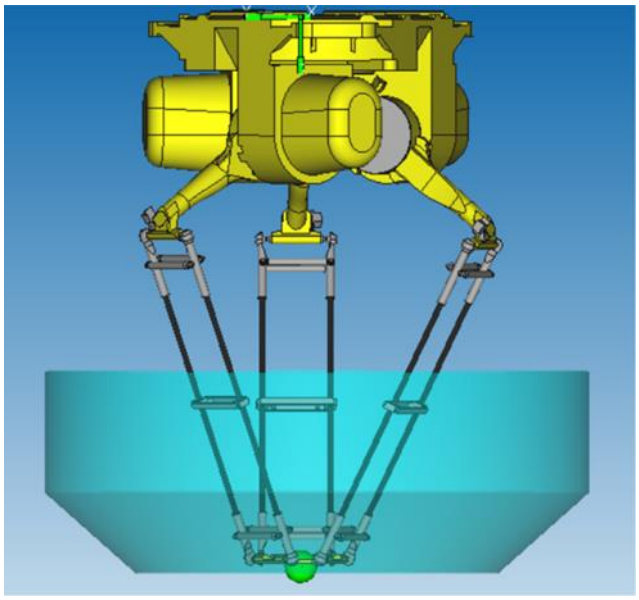


Figure 6.18 - Tool of Delta robot at its lowest position.

The XY plane of the workspace is circle-shaped. So, to complete the calculation of the workspace envelope, we need to split the z length, which stands between the lower and upper limit in 100 samples, and in every sample, we define one-point z. For every z point in a sample, we iterate by increasing the value of x in the inverse kinematics equations beginning from the minimum value of 0 till at least one of the joints exceeds its limits. At that point, we consider the penultimate value of x as the most significant limit in that circle. The same process can be repeated for the y axis.

6.5.6 Limits Check

To avoid a crash when the robot joints exceed their limits, we need to measure the actual position of each joint continuously. If for any reason, it trespasses that limit, the robot will stop emergently.
When working near the joints limits, the software checks if the robot has arrived at the skipping point. It then sends a signal to the robot controller to skip by activating the skip functioning. However, if the robot crosses the working area limits, the software will stop the robot immediately. In our case, if any joint variable θ_i exceeds 80^o or falls below -48^o, then the robot will stop instantly.

When the robot controller receives a skipping signal, it verifies which frame needs to activate the skip function. It stops the robot in its actual position and turns back to the home position to begin a new cycle.

7. Case Study: Experimentation, Results and Perspective

7.1 Introduction

Throughout this chapter, we consider a detailed case study for the developed CT Flex robot while in action. First, we introduce a comprehensive description of the chosen scenario. Next, we discuss how several tasks will be achieved with the help of the software such as defining the reference system for pick and place, the sensing capabilities, the skipping function, and the extra torque detection approach. In addition, we present how to perform the tracing of the Conveyor, defining via point, tuning of parameters, and how to plan the trajectory. After that we illustrate the experimental results. At the end of this chapter, we describe the future perspective, including dynamic modeling and areas of improvement.

7.2 Case Study

7.2.1 Defining Reference System for Pick and Place

In this sub-section, we discuss how to define the pick and place framework. We need to define two working frames: the pick frame and the place frame. With the help of Logix Designer, we start from the robot's world frame to create relative frames. Figure 7.1 shows a snapshot from the software upon derivation of position of a reference frame.

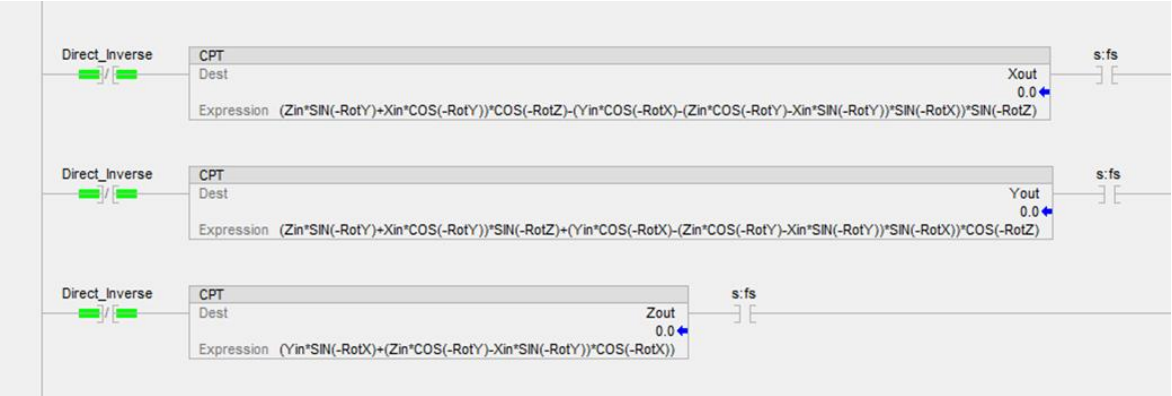


Figure 7.1 - Using Logix Designer to calculate the position of a reference frame.

We start by negating the robot's world frame to turn it upside down, then we assign the relative rotation of each part. In our case, there is no rotation with respect to the X and Y axes, but a rotation of 179.9^{o} with respect to the Z axes of frame R_2 (RotY). After that, we proceed to calculate the

position of the newly defined frame, denoted by $R_2.X_{out}$, $R_2.Y_{out}$ and $R_2.Z_{out}$ respectively using equations 7.1, 7.2 and 7.3 respectively.

$$R_2.X_{out} = \big((X_{in} \cdot \sin RotZ + Y_{in} \cdot \cos RotZ) \cdot \sin RotZ + Z_{in} \cdot \cos RotX\big) \cdot \sin RotY + (X_{in} \cdot \cos RotZ - Y_{in} \cdot \sin RotZ) \cdot \cos RotY \tag{7.1}$$

$$R_2.Y_{out} = \big((X_{in} \cdot \sin RotZ + Y_{in} \cdot \cos RotZ) \cdot \cos RotZ - Z_{in} \cdot \sin RotX\big) \tag{7.2}$$

$$R_2.Z_{out} = \big((X_{in} \cdot \sin RotZ + Y_{in} \cdot \cos RotZ) \cdot \sin RotX + Z_{in} \cdot \cos RotX\big) \cdot \cos RotY - (X_{in} \cdot \cos RotZ - Y_{in} \cdot \sin RotZ) \cdot \sin RotY \tag{7.3}$$

In a similar fashion, we repeat the same calculations for the position of reference frames R_3 and R_4 using equations 7.4, 7.5, 7.6, 7.7, 7.8, and 7.9 respectively.

$$R_3.X_{out} = (Z_{in} \cdot \sin(-RotZ) + X_{in} \cdot \cos(-RotY)) \cdot \cos(-RotZ) - (Y_{in} \cdot \cos(-RotX) - (Z_{in} \cdot \cos(-RotY) - X_{in} \cdot \sin(-RotY)) \cdot \sin(-RotX)) \cdot \sin(-RotZ) \tag{7.4}$$

$$R_3.Y_{out} = (Z_{in} \cdot \sin(-RotY) + X_{in} \cdot \cos(-RotY)) \cdot \sin(-RotZ) + (Y_{in} \cdot \cos(-RotX) - (Z_{in} \cdot \cos(-RotY) - X_{in} \cdot \sin(-RotY)) \cdot \sin(-RotX)) \cdot \cos(-RotZ) \tag{7.5}$$

$$R_3.Z_{out} = (Y_{in} \cdot \sin(-RotX) + X_{in} \cdot \sin(-RotY) + Z_{in} \cdot \cos(-RotY)) \cdot \cos(-RotX) \tag{7.6}$$

$$R_4.X_{out} = (Z_{in} \cdot \sin(-RotY) + X_{in} \cdot \cos(-RotY)) \cdot \cos(-RotZ) - (Y_{in} \cdot \cos(-RotX) - (Z_{in} \cdot \cos(-RotY) - X_{in} \cdot \sin(-RotY)) \cdot \sin(-RotX)) \cdot \sin(-RotZ) \tag{7.7}$$

$$R_4.Y_{out} = (Z_{in} \cdot \sin(-RotY) + X_{in} \cdot \cos(-RotY)) \cdot \sin(-RotZ) + (Y_{in} \cdot \cos(-RotX) - (Z_{in} \cdot \cos(-RotY) - X_{in} \cdot \sin(-RotY)) \cdot \sin(-RotX)) \cdot \cos(-RotZ) \tag{7.8}$$

$$R_4.Z_{out} = (Y_{in} \cdot \sin(-RotX) + X_{in} \cdot \sin(-RotY) + Z_{in} \cdot \cos(-RotY)) \cdot \cos(-RotX) \tag{7.9}$$

Using the above equations, we can deduce the position of the pick frame, denoted by X_{out}, Y_{out} and Z_{out}, according to equations 7.10, 7.11 and 7.12 respectively.

$$X_{out} = R_2.X_{out} + R_3.X_{out} + R_4.X_{out} \tag{7.10}$$

$$Y_{out} = R_2.Y_{out} + R_3.Y_{out} + R_4.Y_{out} \quad (7.11)$$

$$Z_{out} = R_2.Z_{out} + R_3.Z_{out} + R_4.Z_{out} \quad (7.12)$$

Finally, the same procedure could be repeated to obtain the place frame.

7.2.2 Sensing Capabilities

The CT Flex robot is equipped with optical sensors for detection of product packages on the conveyor belt (Figure 7.2). The sensor is registered with the controller and will report a binary status (registration status) of value zero. Once detected, the sensor will trigger informing the controller with the passage of a package by updating its registration status to a value of one, thus arming an event.

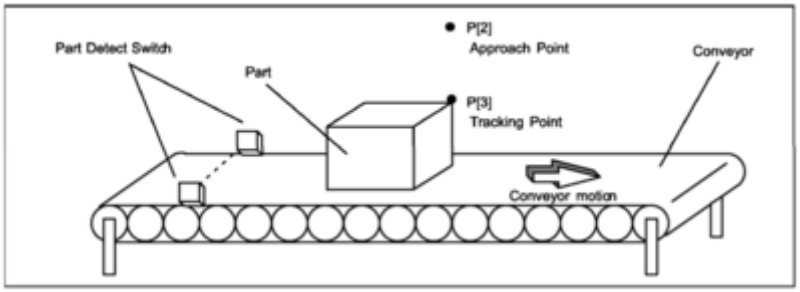


Figure 7.2 - Optical sensors used in CT Flex robot on the conveyor belt.

The controller responds by computing the axis position based on latched encoder count at the same instant when the event occurred and stores it in the associated position variable of the axis data structure. Figure 7.3 shows the registration input of the optical sensor.

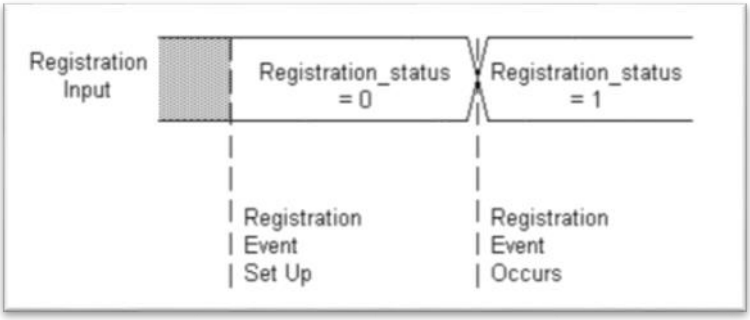


Figure 7.3 - Registration input from the optical sensor.

7.2.3 Performing Skipping Function

The skipping function is very necessary to ensure that the robot doesn't reach to a situation where it might be damaged by self-colliding or colliding with nearby objects while trying to reach out to product packages on the conveyor belt.

To achieve that, we need to constantly monitor the position of the end-effector while the robot is in operation. This could be done by invoking the kinematic equations of the robot and use the joint variables θ_1, θ_2 and θ_3 to calculate the position of the end-effector.

If the calculated position of the end-effector is not within the limits (derived as explained in section 6.4.1 Limits Check), then this means that the robot is entering a skipping zone and in this case the controller will command the robot to not proceed at all and halt the current operation. Figure 7.4 summarizes the algorithm behind the skipping function.

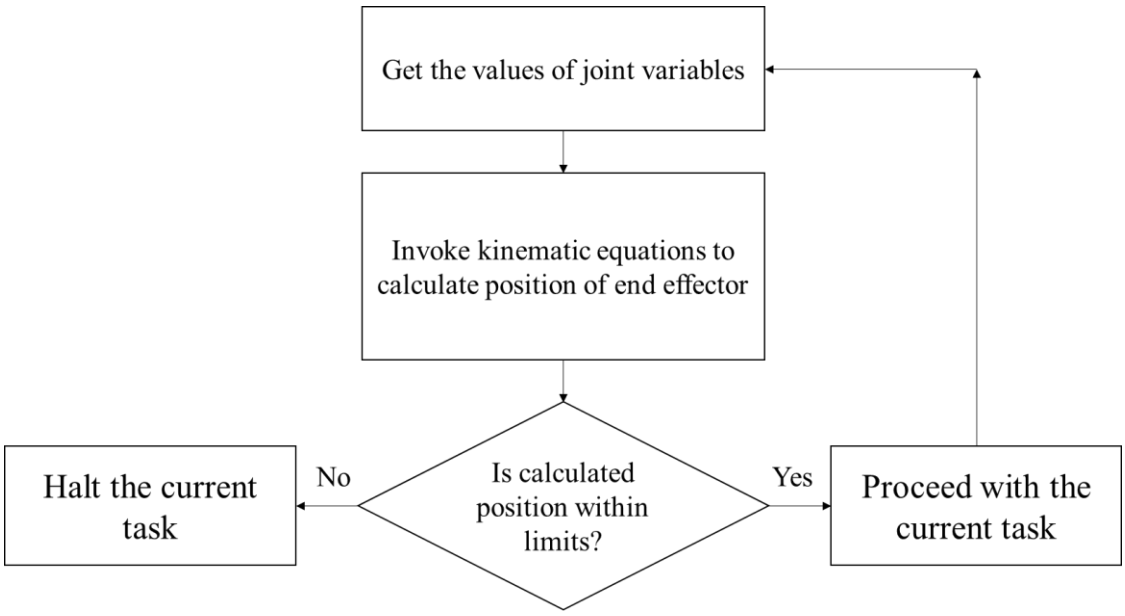
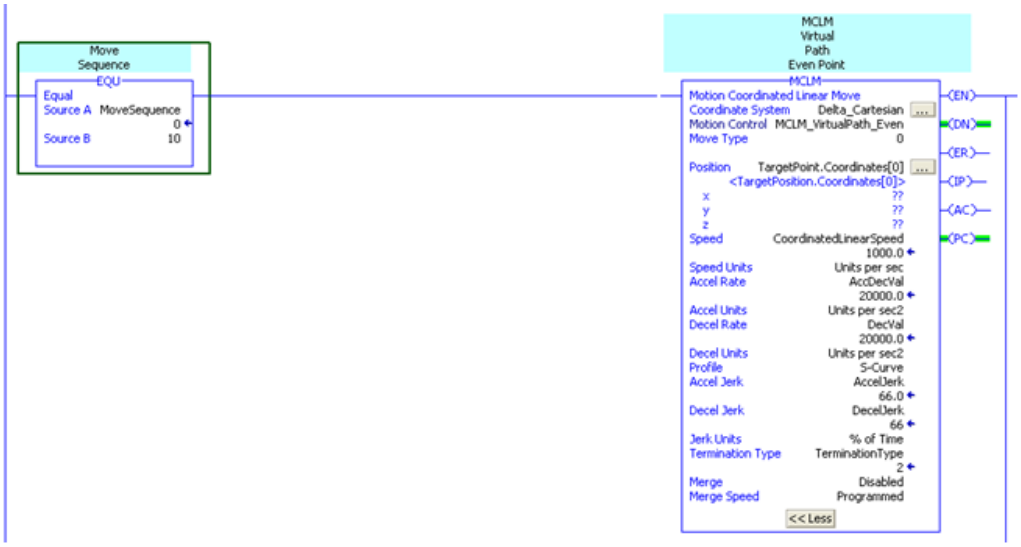


Figure 7.4 - Algorithm of skipping function.

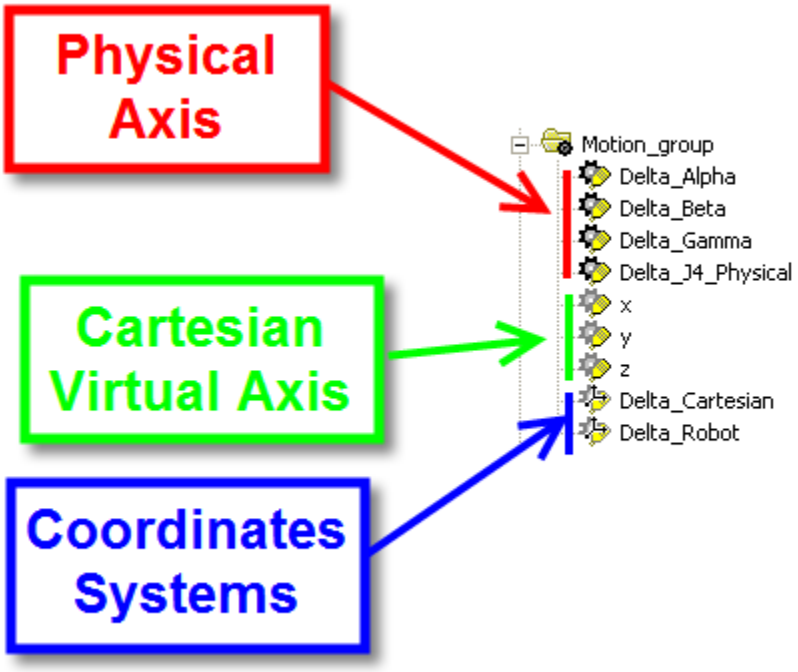
7.2.4 Analysis of Robot Dynamics using Logix Designer and Motion Analyser

In this section it is shown how the Dynamics of the Delta robot has been evaluated, using the Motion Analyzer software, a tool provided by Rockwell Automation to support machine designers in the choice of motors from the Rockwell catalogue [18]. The Dynamics of a Delta

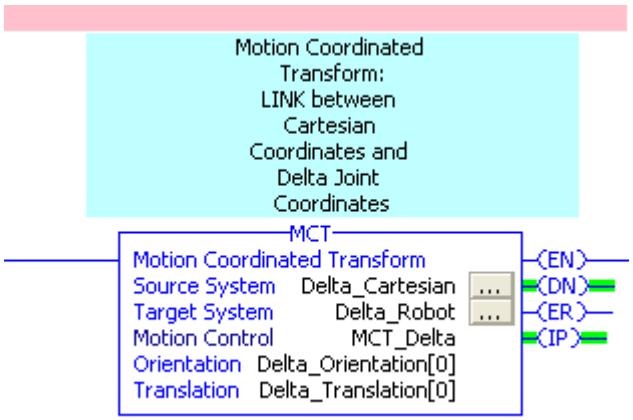
Robot Application can be evaluated only on a Motion Profile. The easiest way, considered here, is to build a Delta Robot Kinematics using the pre-defined modules in Logix Designer and use the motion coordinates instruction to create a motion profile and build a sequence of movement using MCLM instructions (Figure 7.5).



In this situation we have two Coordinates System:



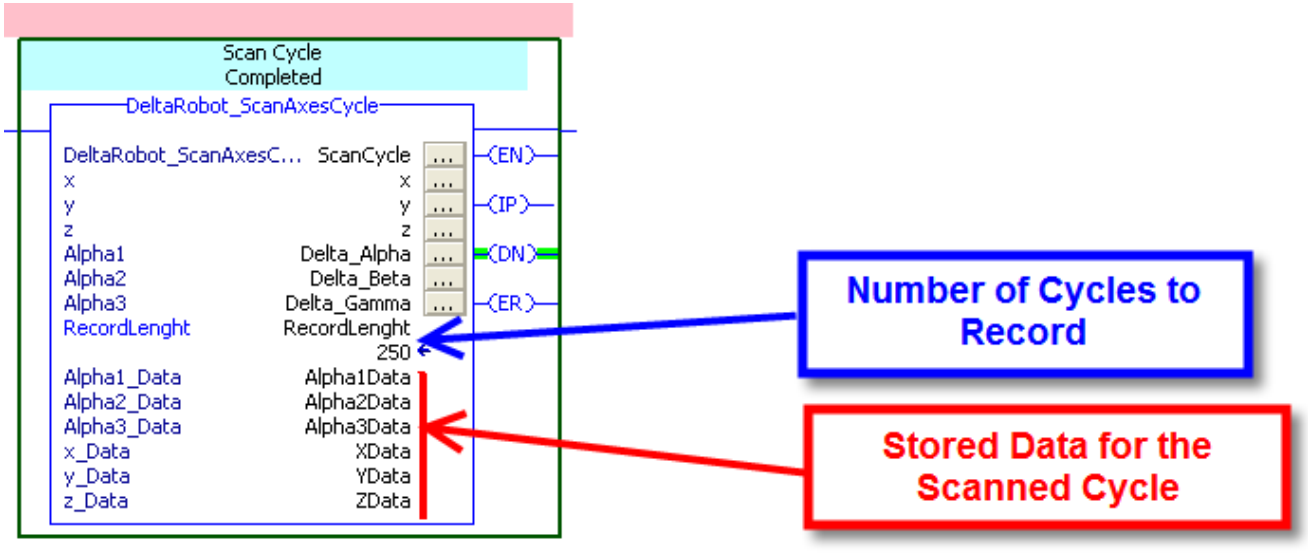
by means of the MCT instruction they are linked Together.



This is what we need to **handle easily generalized coordinates**.

$(\ldots \{q_i\} = \{x, y, z, \alpha_1, \alpha_2, \alpha_3\} \ldots)$

By means of two Add-On Instruction we **RECORD** x,y,z, Alpha, Beta, Gamma Actual Positions, Actual Velocity and Actual Acceleration. Knowing how Actual Velocity and Actual Acceleration are built we could have done it ourselves, but they are ready, so the recording of the DATA must be synchronous with Motion Planner, so it is a good advice to place the record instruction in Motion Synch Task. Mind to be aware **not to have overlap** otherwise you'd think you're sampling at a certain time-rate but with overlaps the effective rates would be at least the double altering all the results.



Remember that units of measurement MUST be radians and Metres for angular and linear displacements. Be sure to transform Engineering Units so to fit those Values:

```
x_Data[0,Index] := (x.ActualPosition /1000);        // Linear Data Must be Expressed as metres, m/sec, m/sec^2
x_Data[1,Index] := (x.ActualVelocity /1000);
x_Data[2,Index] := (x.ActualAcceleration /1000);

y_Data[0,Index] := (y.ActualPosition /1000);
y_Data[1,Index] := (y.ActualVelocity /1000);
y_Data[2,Index] := (y.ActualAcceleration /1000);

z_Data[0,Index] := (z.ActualPosition /1000);
z_Data[1,Index] := (z.ActualVelocity /1000);
z_Data[2,Index] := (z.ActualAcceleration /1000);
```

End Effector Position is expressed in Mm, it MUST be transformed in Meters

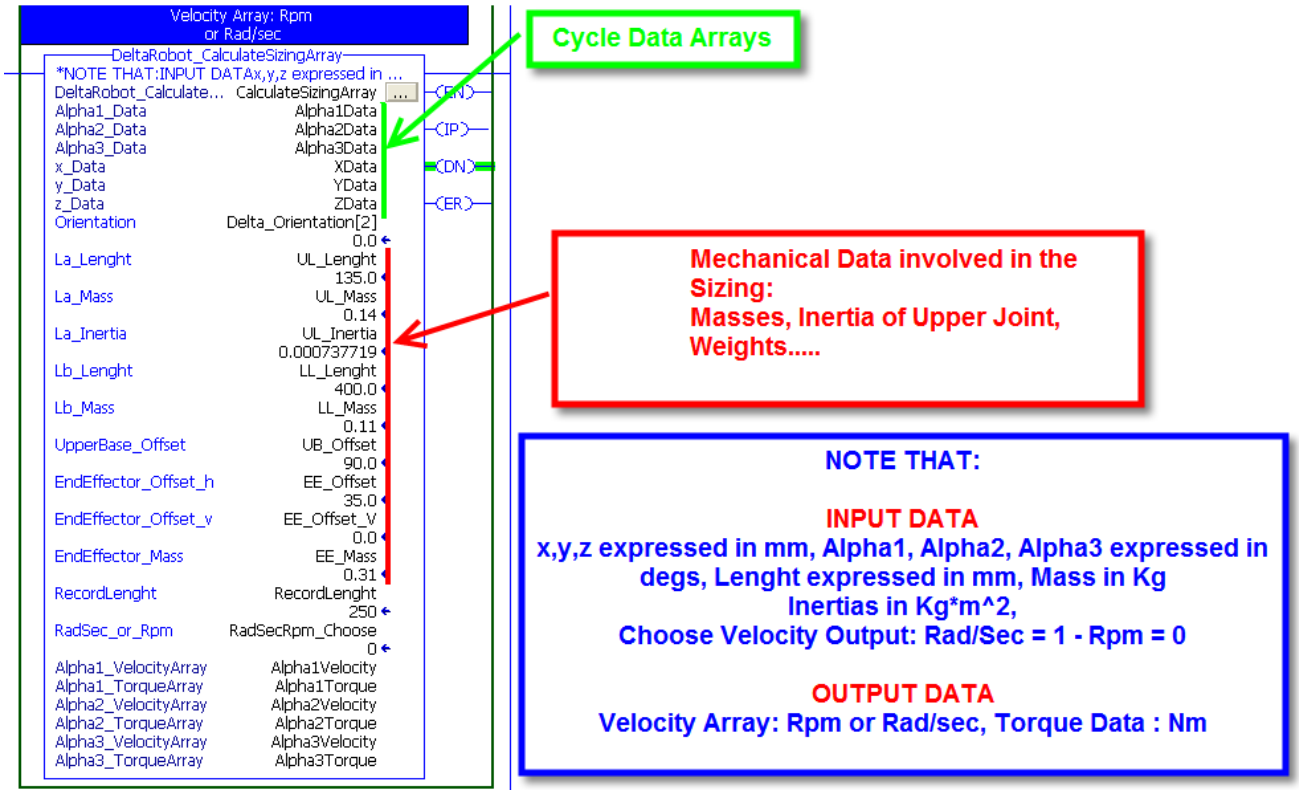
```
Alphal_Data[0,Index] := RAD(Alphal.ActualPosition);     // Rotary Units MUSt be expressed in Radians, Rad/sec, Rad/sec^2;
Alphal_Data[1,Index] := (Alphal.ActualVelocity*Pi)/180;
Alphal_Data[2,Index] := (Alphal.ActualAcceleration*Pi)/180;

Alpha2_Data[0,Index] := RAD(Alpha2.ActualPosition);
Alpha2_Data[1,Index] := (Alpha2.ActualVelocity*Pi)/180;
Alpha2_Data[2,Index] := (Alpha2.ActualAcceleration*Pi)/180;

Alpha3_Data[0,Index] := RAD(Alpha3.ActualPosition);
Alpha3_Data[1,Index] := (Alpha3.ActualVelocity*Pi)/180;
Alpha3_Data[2,Index] := (Alpha3.ActualAcceleration*Pi)/180;
```

Axes Position is expressed in Degs, It MUST be transformad in Radians

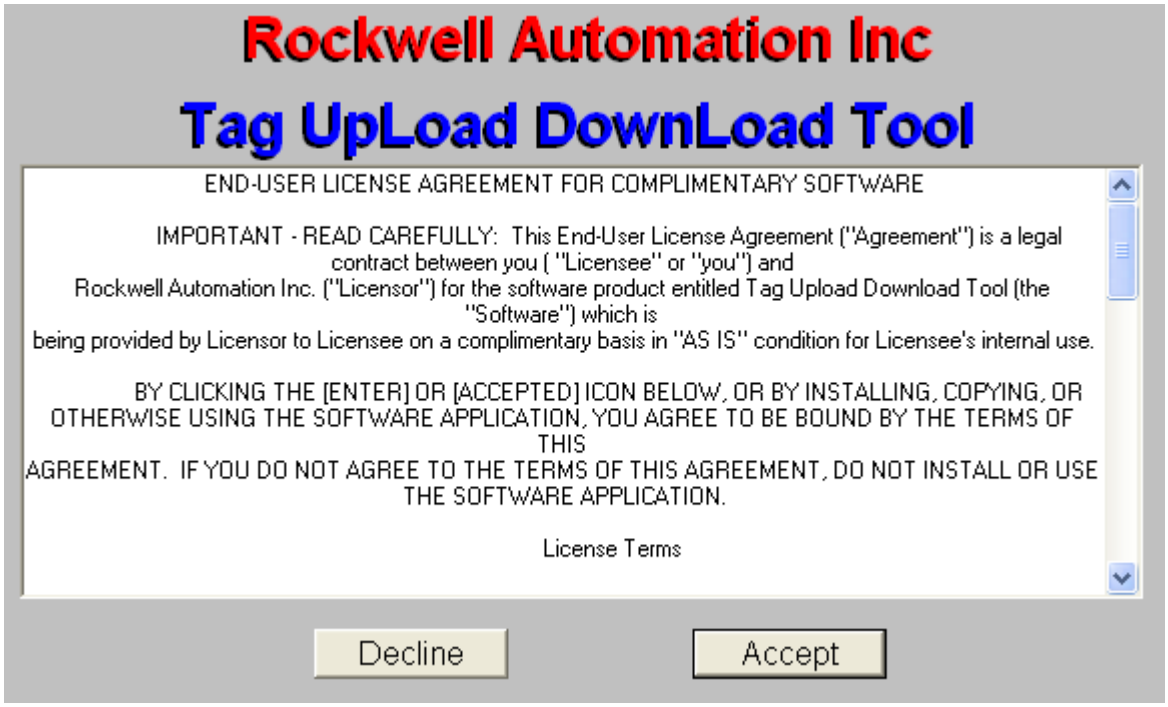
Once DATA has been collected they must be **PROCESSED.** The second Add-On Instruction reads the Position, Velocity and Acceleration Data from the Recorded Arrays and elaborates them accordingly to the Equations written previously:

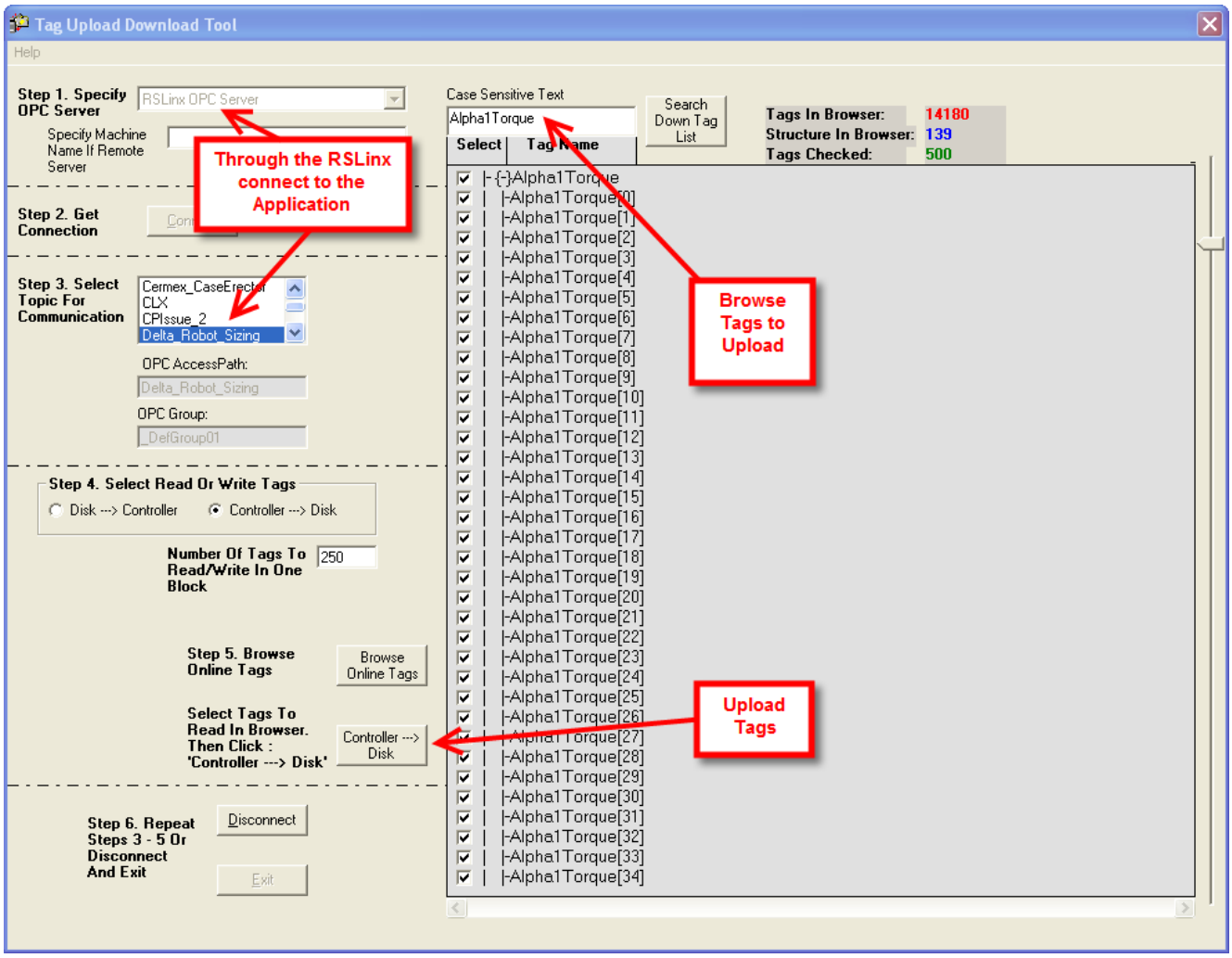


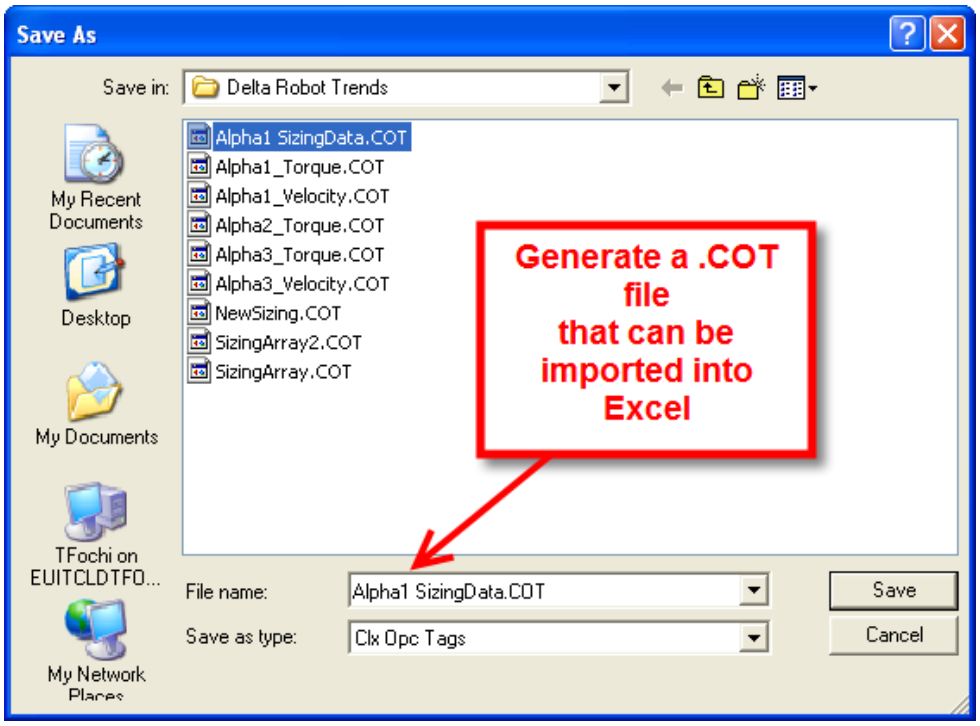
This Instruction uses all Input Data to build a Velocity-Torque Output arrays suitable for Motion Analyzer. Due to the Complexity of the calculation the execution of this instruction is quite slow.

(Slow Task Scan time arise from 6ms to 22ms while the instruction is running). For this reason it has been placed in the slowest task.

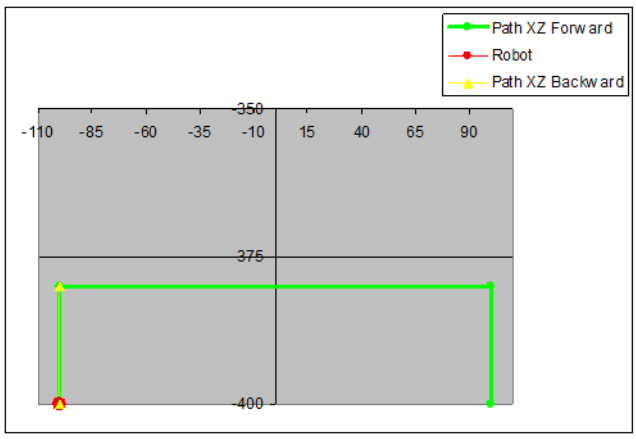
The Output data of the Instruction are six arrays (velocity and Torque for each Robot Axis), that can exported by means of “Tag Upload Download Tool”

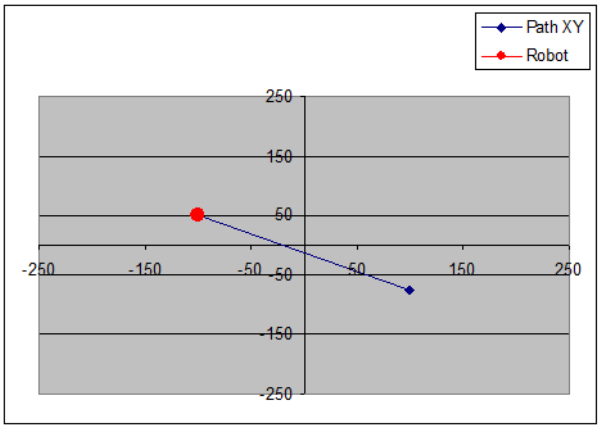






The Cycle to Size is:

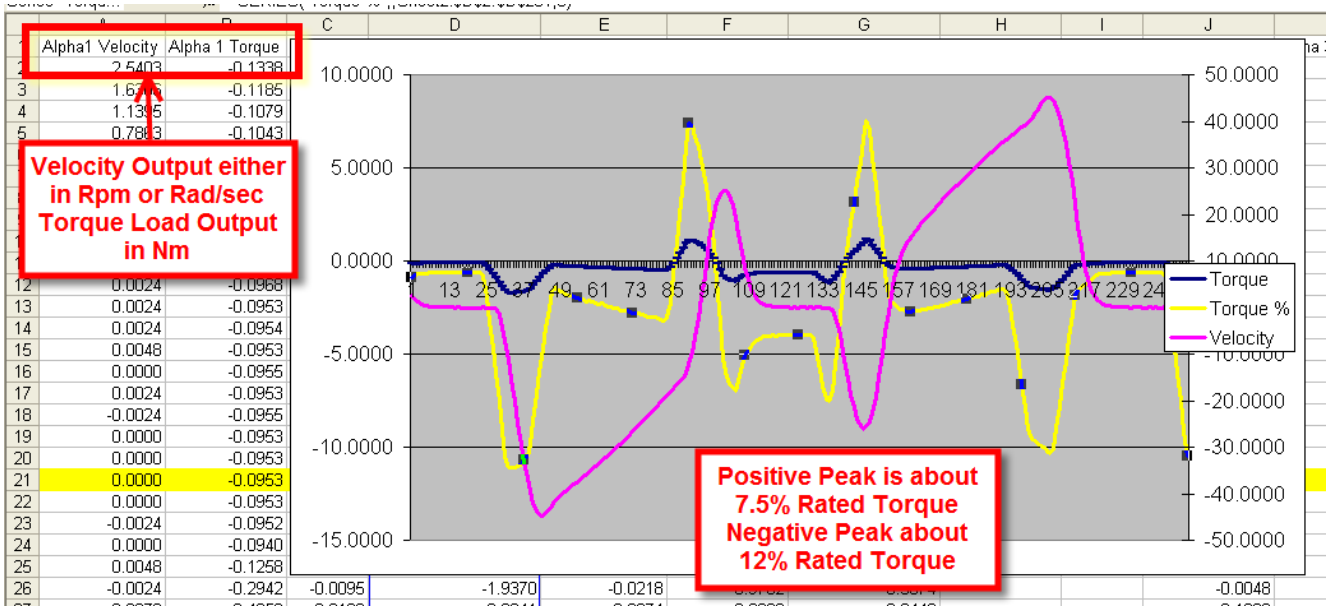




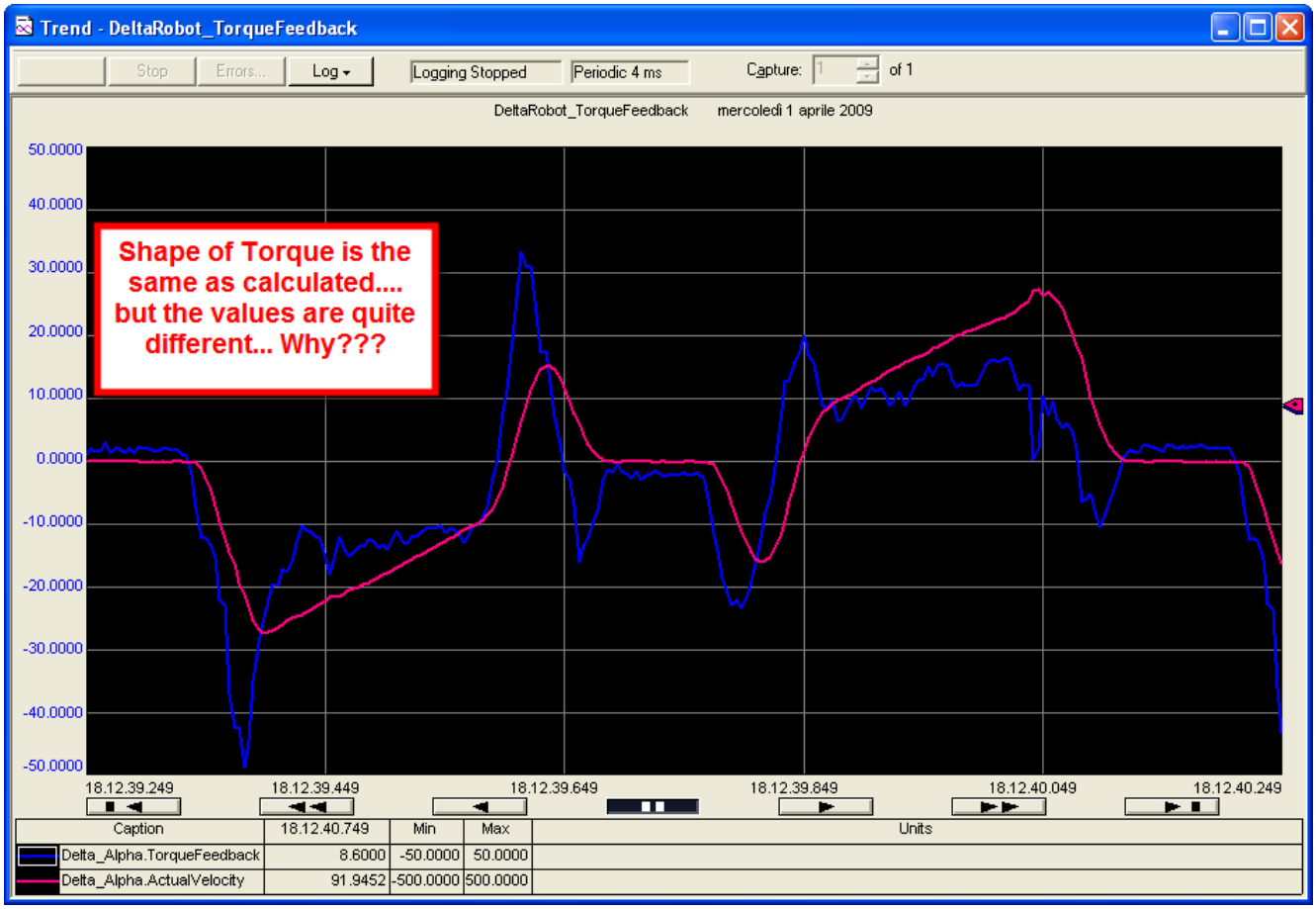
Pick Point: $\begin{cases} X = -100 \\ Y = 50 \\ Z = -400 \end{cases}$ Place Point $\begin{cases} X = 100 \\ Y = -75 \\ Z = -400 \end{cases}$

Intermediate Points: $\begin{cases} X = X_{Pick}, X_{Place} \\ Y = Y_{Pick}, Y_{Place} \\ Z = -380 \end{cases}$

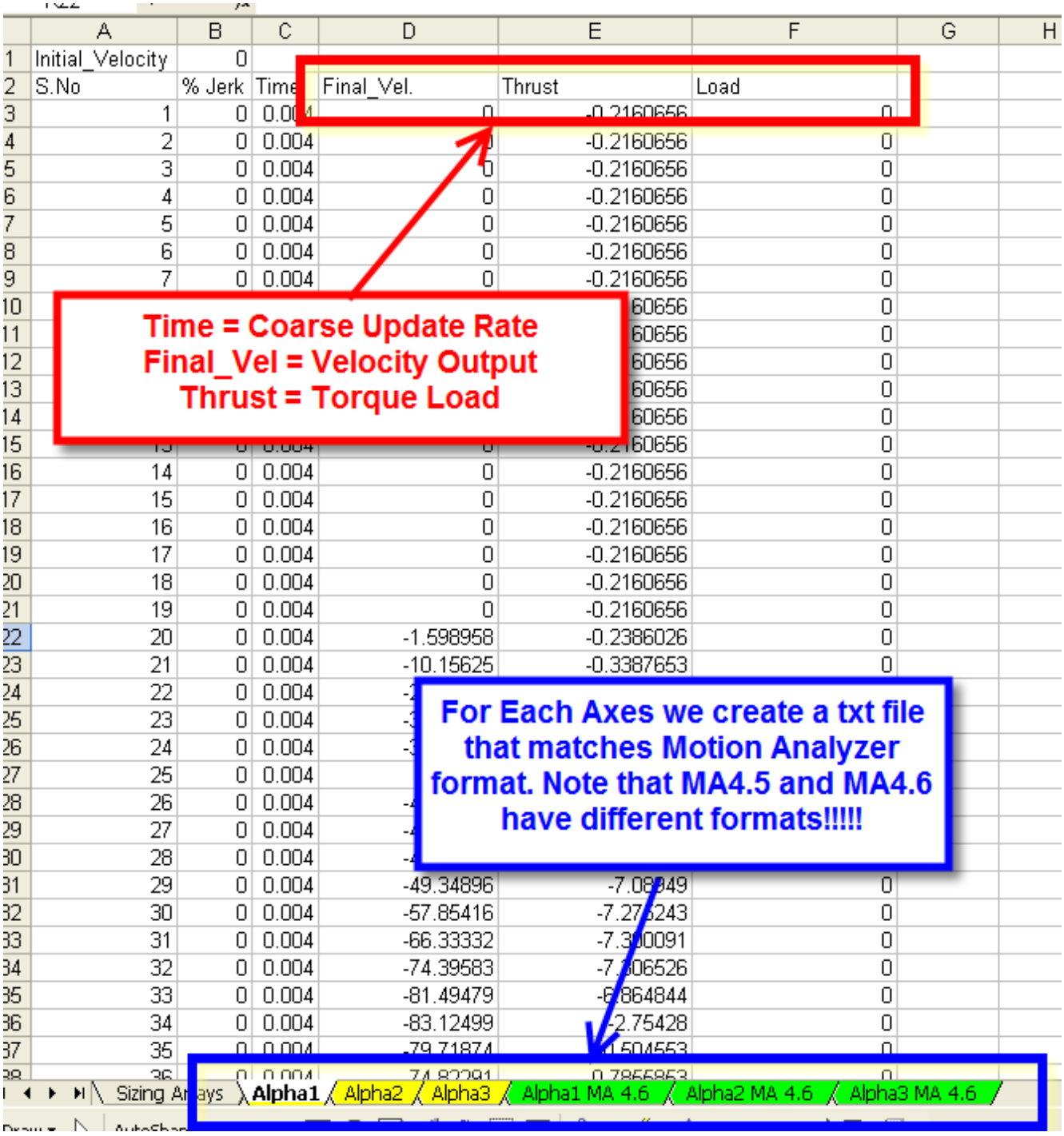
With a desired performance of about 60 Ppm. In the figure below the theoretical Torque profile for axis Alpha1.



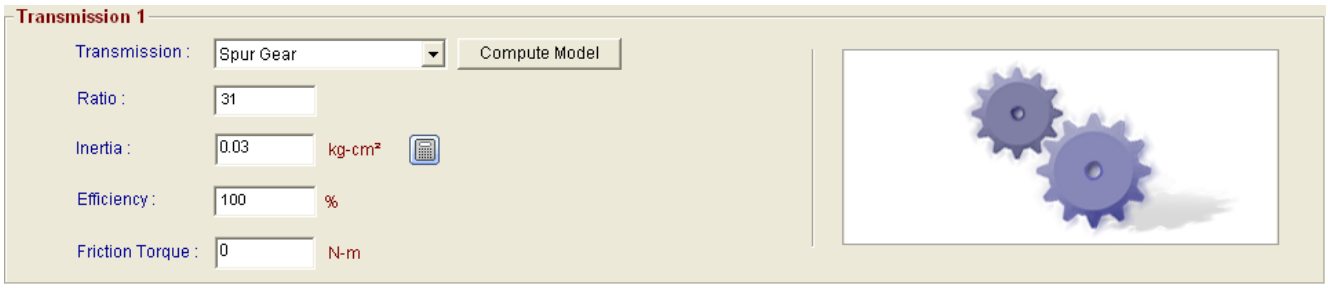
First thing we notice is that Motor Torque Feedback values are quite different from Motor Estimated Torque values (of course from Nm you need to translate in Torque % to compare them).



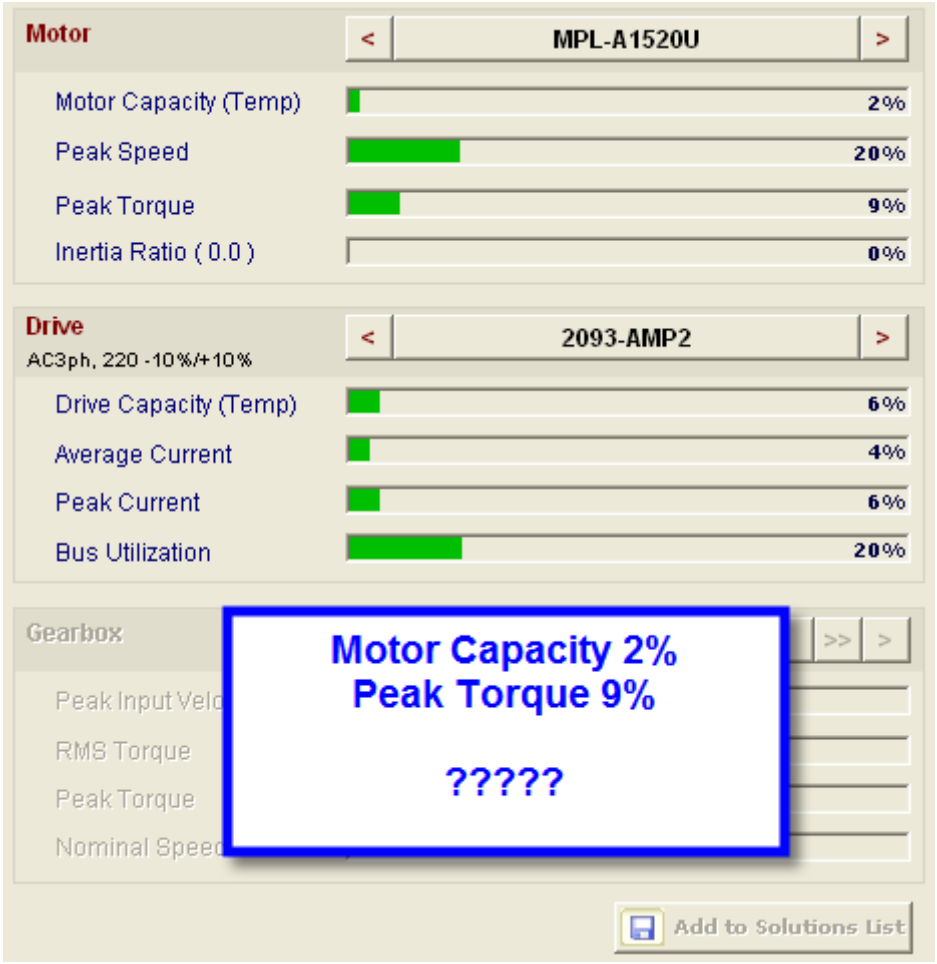
Now we can create the **Motion Analyzer File**:



If we model the Gearbox as a Spur Gear with a given Inertia and a given Efficiency, as stated in the Gearbox datasheet (Gearbox is an APEX AD047-PO Planetary in Line, 31:1 Ratio, 0.03 Kg*Cm2 Inertia).



The results are still quite far from reality…



Note that Peak Torque in Motion Analyzer is expressed as percentage of Peak Torque. So, if we want to translate into % of Rated Torque we need to do:

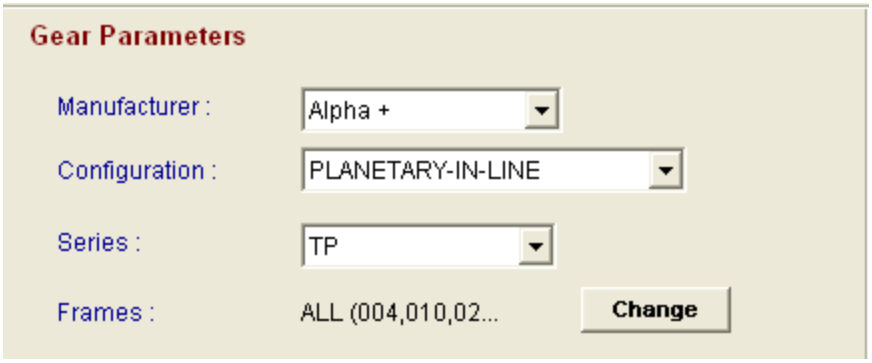
$$T_{\%ofRatedTorque} = T_{\%ofPeakTorque} * \frac{T_{Peak}}{T_{Rated}}$$

If this case

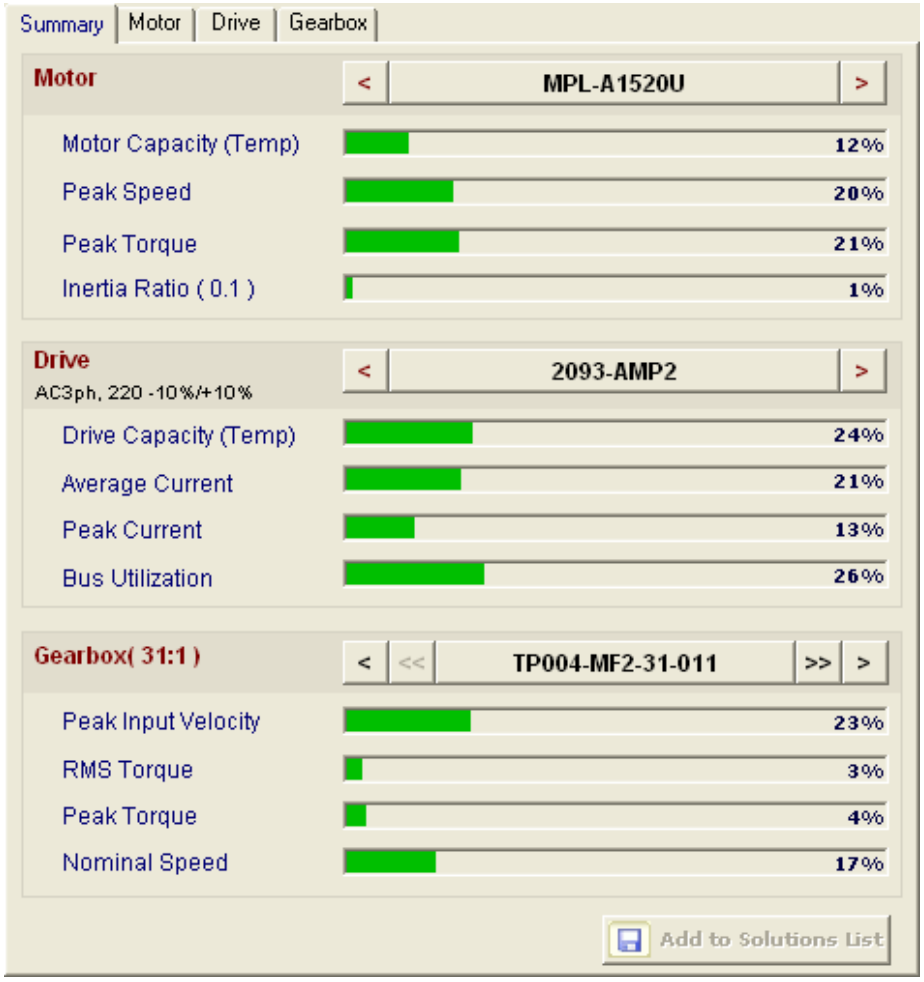
$$T_{\%ofRatedTorque} = T_{\%ofPeakTorque} * \frac{T_{Peak}}{T_{Rated}}$$

$$T_{\%ofRatedTorque} = 9\% * \frac{1.3929Nm}{0.49667Nm} = 25.2\%$$

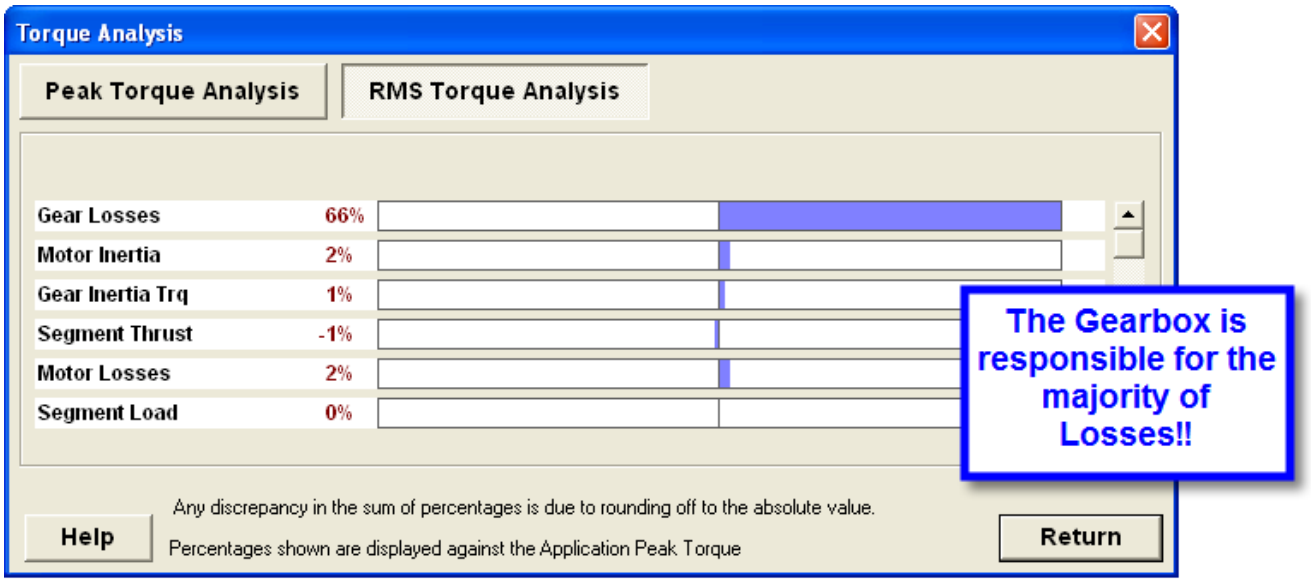
But if we insert an Alpha Gearbox the most similar to the APEX one:



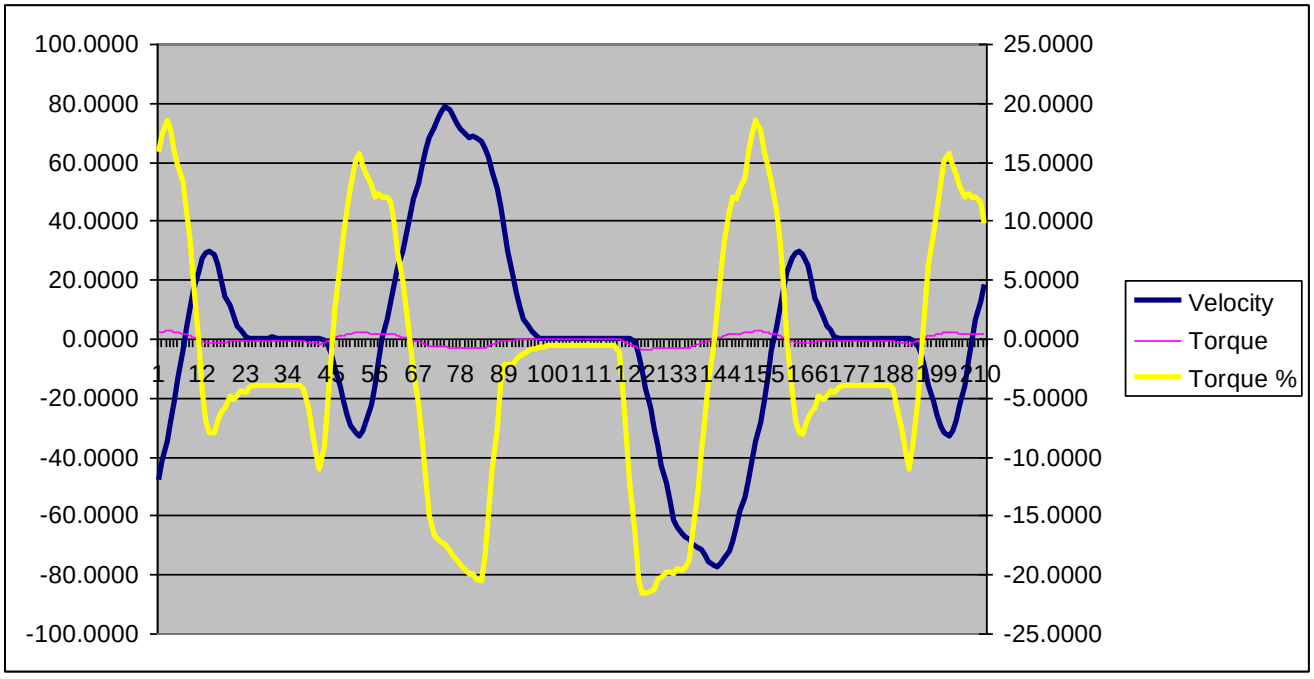
Results are:



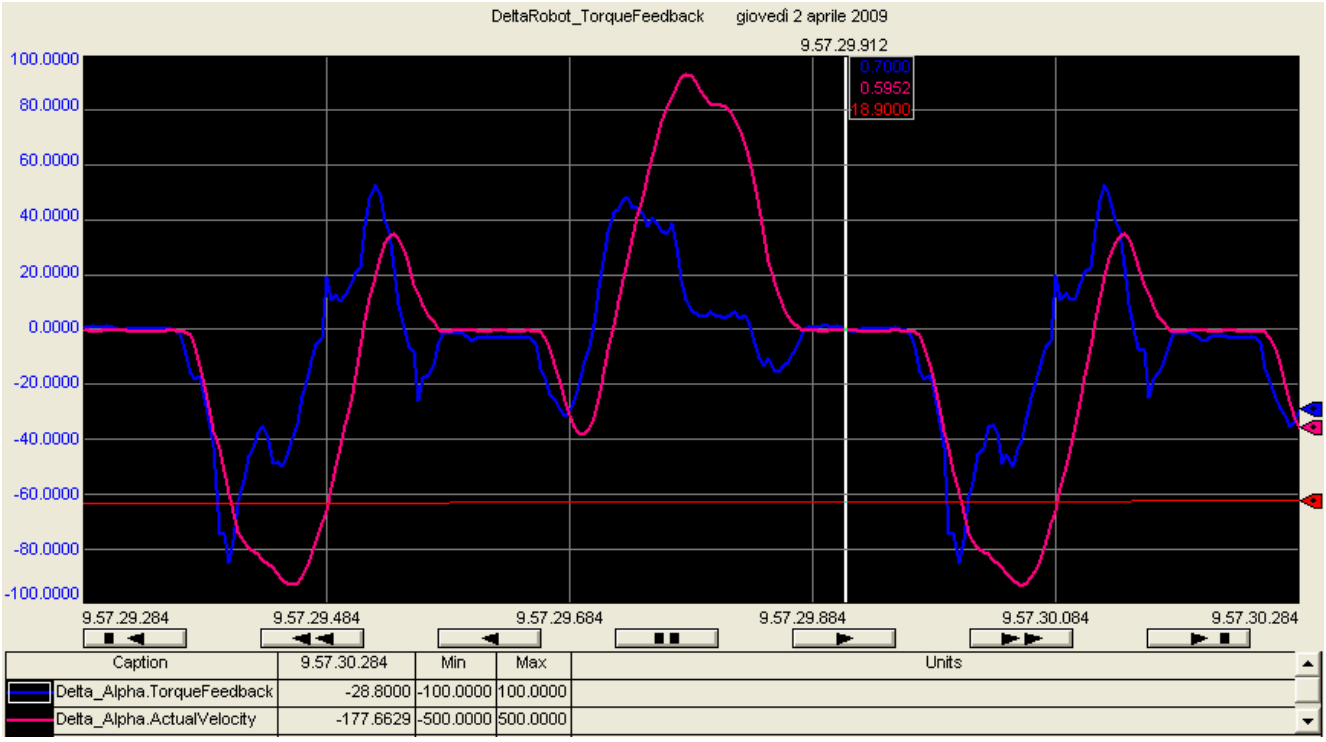
A lot more similar to what we saw in the Torque Feedback Trend.



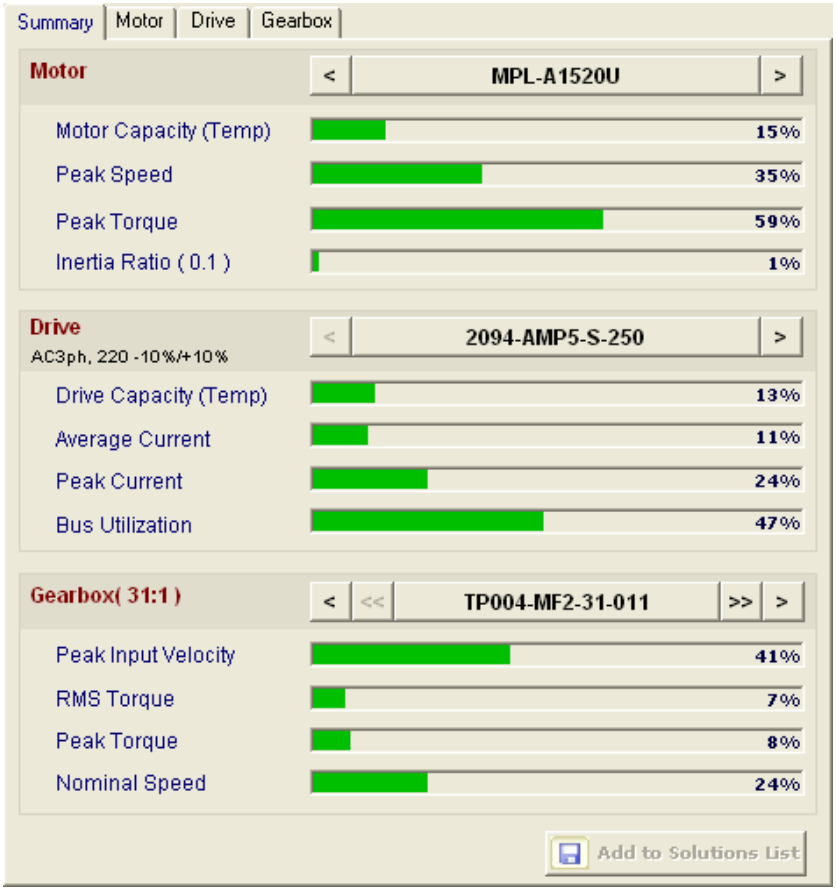
It seems, than, that in the sizing of this type of application it is very important to understand what is the role of the Gearbox and how we can correctly estimate its weight in the total losses. Let's move up the desired Production Rate to about 110 ppm, and same path as before. Theoretical Torque profile for Axis Alpha1 is:



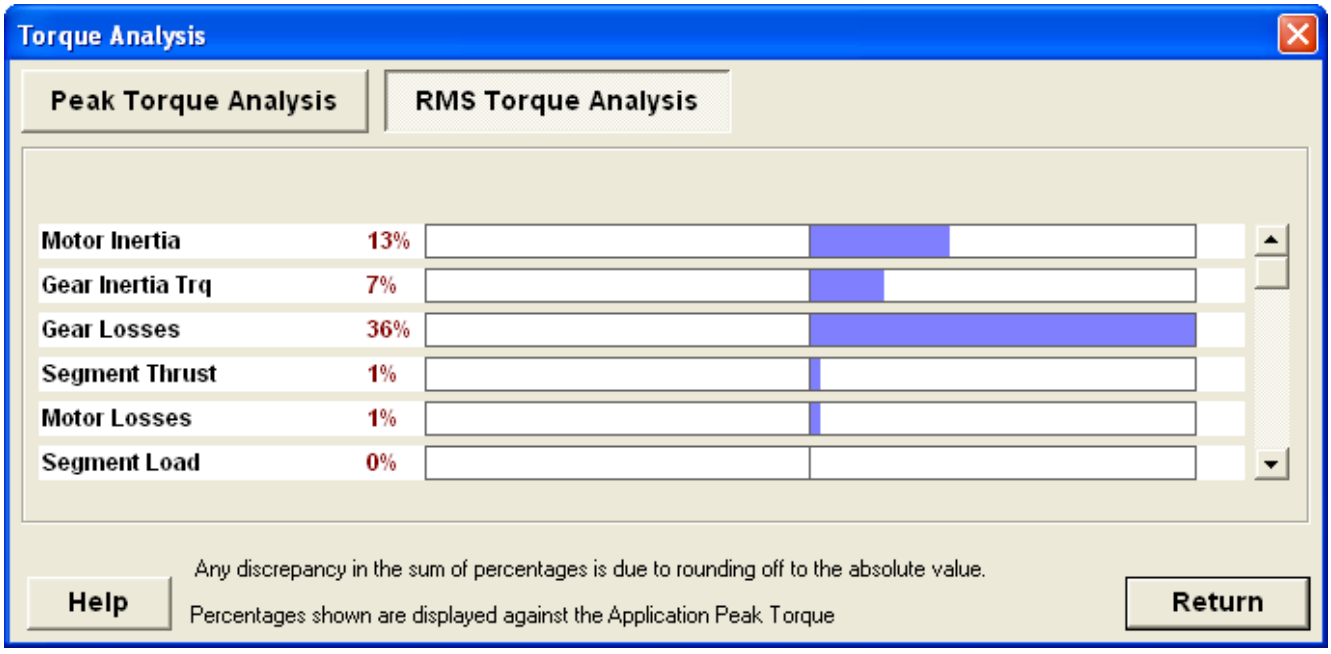
Torque Feedback recorded:



Placing the curves into Motion Analyzer:



The weight of the Gearbox, as then confirmed from motion analyser is decreasing:



It has been decreased from 66% to 36%. We can easily imagine that adding a full load or a heavier end-effector will keep on pushing the application this way. In these cases the impact of the gearbox reduces, and it is no more so important to model it perfectly.

7.2.5 Conveyor Tracking Frame

Conveyor tracking is the process of aligning and then controlling the conveyor belt so that it maintains a desired path. To allow that, we need to properly setup a tracking frame. When setting a tracking frame, we need to have a fixture carried on the conveyor. Any fixture may be used if it has a part that the robot can touch up accurately with the tool center point.

For our case, we consider a line conveyor. Using Logix Designer, we start by adding as many conveyor stations as the number of robots that will operate on the conveyor. In the setup screen for each conveyor station, we select a robot, tracking schedule number, and encoder number. Next, we place a fixture that the robot can touch up with the tool center point in the most upstream area of the conveyor. Now we move the conveyor until the fixture reaches the upstream area within the robot operation range, and then stop it. We jog the robot to touch up the fixture with the tool center point. Here, the robot touches up the dot located at the intersection of large dots arranged in the shape of the letter L on the calibration grid (Figure 7.5). Here we mark that point in software.

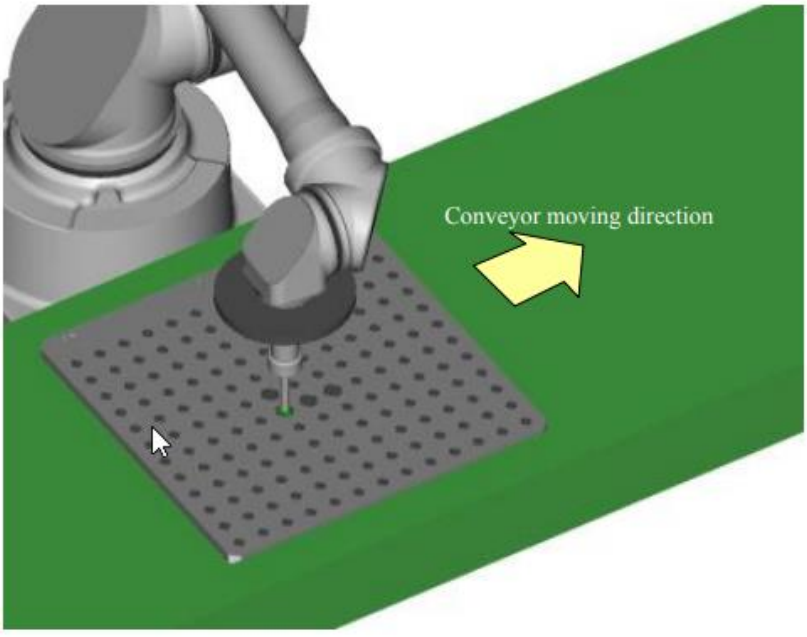


Figure 7.5 - Fixture on conveyor belt during setup of conveyor frame.

Next, we move the robot to a position where the tool center point does not interfere when the conveyor is moved. Now move the conveyor so that the fixture comes to the downstream area within the robot operation range and then stop it. Then, we jog the robot again to touch up the same part of the fixture that you touched up with the TCP earlier. Also we mark this point in the software.

After that, we jog the robot upward (or away from the conveyor surface). Then we jog the robot to the left as much as possible (at least 10 mm) relative to the conveyor moving direction without moving the conveyor. Then we jog the robot downward (or toward the conveyor surface) to touch up a point of the fixture and mark that point also in software. When doing this, we make sure that the tool center point is accurately positioned on the plane surface of the fixture (Figure 7.6). This completes the tracking frame setting procedure for the robot, and a tracking frame and scale have now been set.

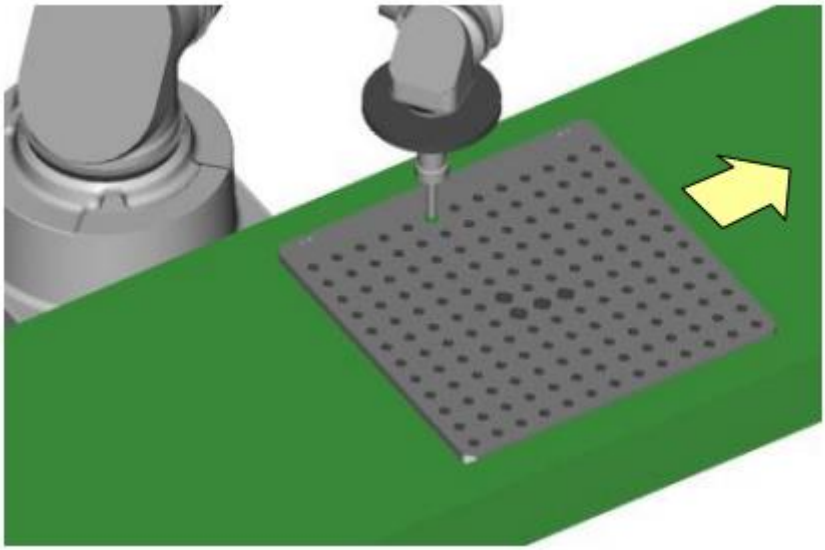


Figure 7.6 - Jogging the robot towards the conveyor surface.

7.2.6 Path Planning

In this section, we describe how the path planning mechanism is done for the CT Flex robot. We will begin discussing the process of moving from a fixed point P1 to another fixed point P2 within the same reference frame A. Moreover, we take into consideration that a gripper might be attached at the end effector which would result in a rotary motion (revolute joint).

The robot will begin from an initial point P1, which is practically the robot's actual position and orientation. The position and orientation of this point P1 can be directly calculated from the joint variables using forward kinematics equations. Here, we need to specify the rotation matrix which

conveys the orientation at destination point P2 with respect to orientation at initial point P1. This can be done by virtually attaching another frame B at point P2 and describe the orientation of this frame B with respect to reference frame A, denoted by ${}^{A}_{B}R$. Hence, the description (position and orientation) at point P2 can be obtained using the following homogeneous transformation equation:

$$ {}^{A}P_2 = {}^{A}P_1 + {}^{A}_{B}R.\,{}^{B}P_2 $$

Note that for the case of pure translation, where no revolute joint exists at the end-effector, the rotation matrix ${}^{A}_{B}R$ is simply an identity matrix and we can directly coordinates of both points P1 and P2.

To achieve that in software, we Use the MCPM instruction to start a multi-dimensional coordinated path move for the specified Primary axes (X, Y, Z) and orientation axes (Rx, Ry, Rz) of a Cartesian coordinate system. We use this instruction to program Cartesian moves on robots with orientation control. Figure 7.7 shows how to assign a target position in the Logix Designer software.

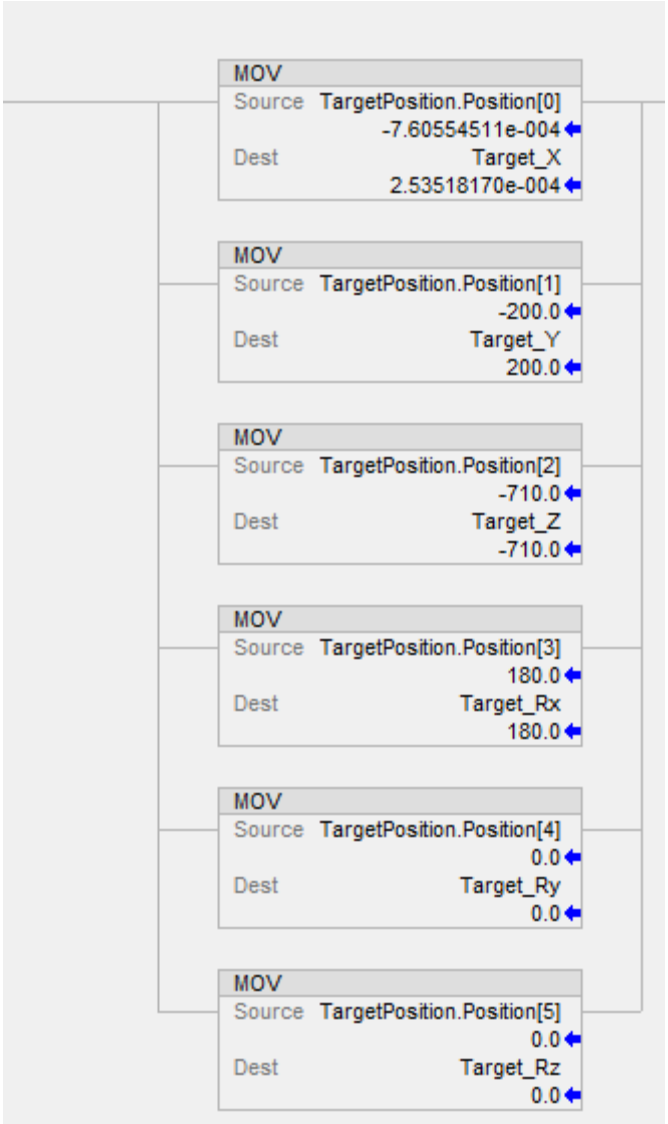


Figure 7.7 - Assigning target position in Logix Designer software.

Figure 7.8 shows how to define the rotation matrix in the software.

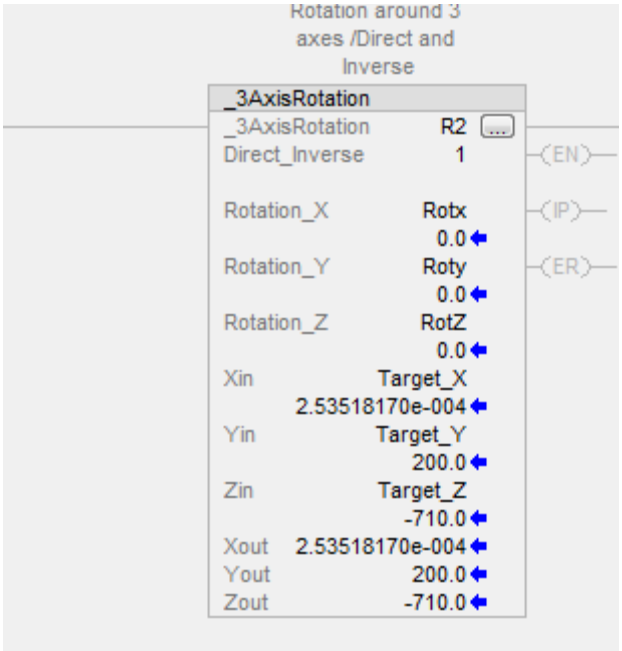


Figure 7.8 - Defining rotation matrix in Logix Designer software.

On the other hand, the robot can move from an initial point P1 in a work frame A to a destination point P2 in another work frame B (note that these points can be fixed point or moving points). The first point P1 represents the actual position of the robot in a work frame A, already chosen within the software. The frame B of the destination point P2 must also be defined in the software. We commence by converting the description of the destination point P2 into the initial work frame A and then and then apply a motion instruction. Figure 7.9 shows how to convert a given position from one frame to another within the software.

Figure 7.9 - Converting a given position from one frame to another in Logix Designer.

7.2.7 Pick and Place Cycle

The pick and place cycle explains how the CT Flex robot performs in production lines where extensive pick and place operations are performed. The very first step is to define the frames of the pick and place on the conveyor belt as described previously in this chapter. At start, the end-effector of the robot is placed directly above the picking place / zone. When the optical sensor detects the passing of an object, it triggers an event to the controller. At this moment, an encoder placed at the conveyor belt will measure the distance travelled by the object since it has passed by the upstream and reports this distance continuously to the robot controller. This means that the robotic system knows the instantaneous position of the object on the conveyor now with respect to the conveyor frame and it will lock its end-effector just above the object to be picked with the help of a camera.

The robot will determine the position of the end-effector using the joints values in the forward kinematics equations and will check if this position is within the acceptable limits to proceed with the picking, otherwise, a skipping function is activated. If within the limits, the robot will perform the conversion of object position from conveyor frame to end-effector frame and then performs the motion instruction. Note that the robot will follow smooth paths during picking and place operations by including intermediate via points.

Now the robot is ready to activate the gripper for picking. To go to the place zone, the robot will convert the position of placing defined previously according to conveyor frame into end-effector frame and then issue motion instruction to move towards the place point. When it reaches there, the gripper is deactivated to release the object in the proper place. At that point, the robot might either go to the picking point to wait for another object or might return directly to lock its end-effector over an object in the quay zone. Figure 7.10 shows the algorithm used for pick and place operations.

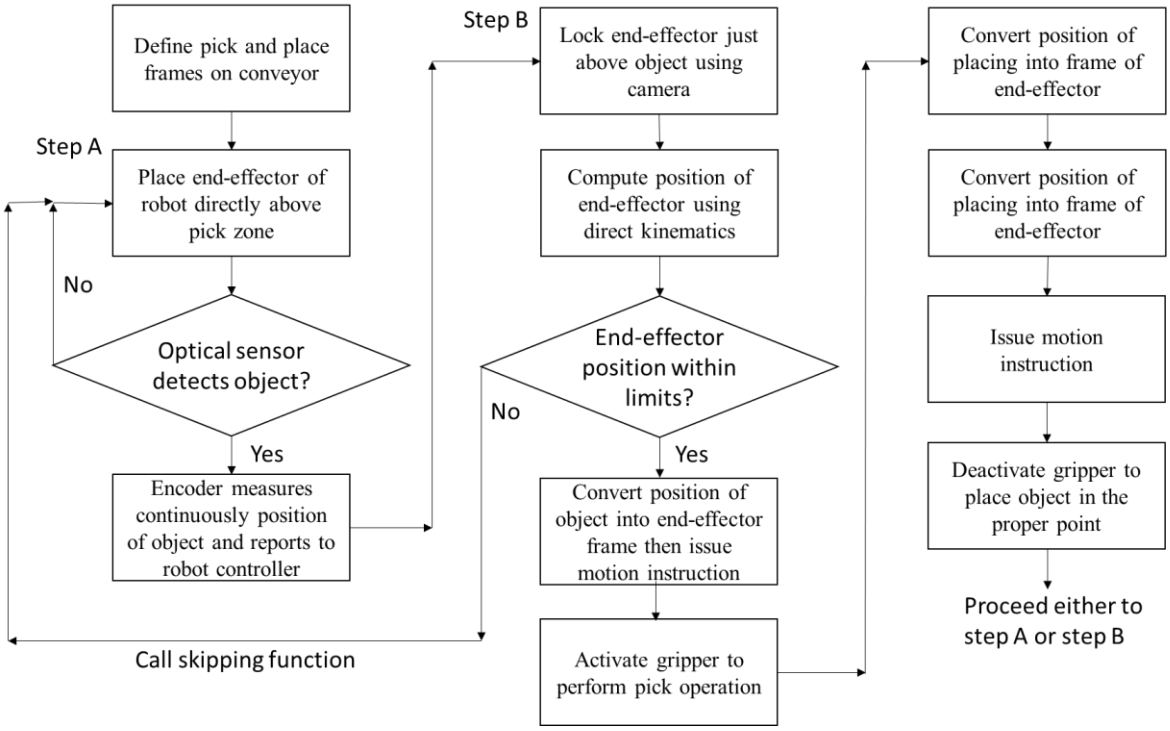


Figure 7.10 - Algorithm for pick and place cycle.

7.3 Experimentation Results

As a result of the experimentation done, our robot is able to handle a payload up to 20 kg, which was not possible with the available commercial delta robots. This higher payload permits to enlarge the

gripper and increase the vacuum part, consequently increasing the number of picked products per minute. In addition, at payload, the robot achieved around 90 percent of the target.

In terms of cost, the design of the CT Flex permits running pick and place production lines at cheaper costs thanks to its modular design.

7.4 Future Perspective

In the fast development of the 4.0 digital revolution, clients' demands for new technologies are constantly increasing. Having a new robot where we can apply plenty of new technologies, the performance is vastly improved while allowing remote monitoring of machine state. In our case, we are able to estimate the quantity of every pick and place, the waiting time, and errors. Moreover, real-time visualization permits anticipation of many problems in advance. Hence, the service department can be fully alerted for maintenance and organization of periodic check-up plans, which is a fundamental point highly demanded by the clients in the market.

Furthermore, multinational companies are always demanding homogeneous production lines composed of repeated / cascaded modules to avoid a complicated interface between the operator and the machine and to reduce maintenance, operational and upgrade costs.

On the other side, we should also reflect on constantly developing the machine's performance, mainly by increasing the speed of the whole production line, producing more boxes per minute, and performing more pickings after expanding the infeed product frequency. In addition, using advanced grippers permits manipulating larger payloads. With CT Flex, we expect to double the actual multi pick used in the existing traditional commercial robot.

The target applications of this robot will mainly be in the industry of food and beverages such as biscuits, chocolate, ice cream, sticks, and cones where the multi-picking functionality is applied. The perspective is to sell the CT Flex robot as a standalone solution, with unique features and user-friendly interface and capabilities of integration with other existing robotic systems.

7.5 Conclusion

Throughout this chapter, we have seen the CT Flex robot in a real life pick and place application. We went through a detailed case study showing how we installed and configured our robotic system. We

started by defining reference system for pick and place operations. Then we defined the sensing capabilities of the robot and the skipping function. After that, we demonstrated how to configure the robot dynamics in software. We defined the tracking frame and discussed path planning. Finally, we explained the algorithm used for the pick and place cycle.

General Conclusion

For many years, the company used the commercial Delta robot in the machine of pick and place, especially the Fanuc type. During these years, and as the optimization of these machines' performances evolved, a new development in the multi-robot systems applied, which is a dynamic boundary. In the specification, the boundaries of the working area changed dynamically based on the lanes of the product chosen to withdraw. Starting from the broadest up and then basing it on the number of the narrow lane respecting the limits of work area, the same goes for the downstream. Figure A shows the gained areas by applying the dynamic boundaries of upstream and downstream.

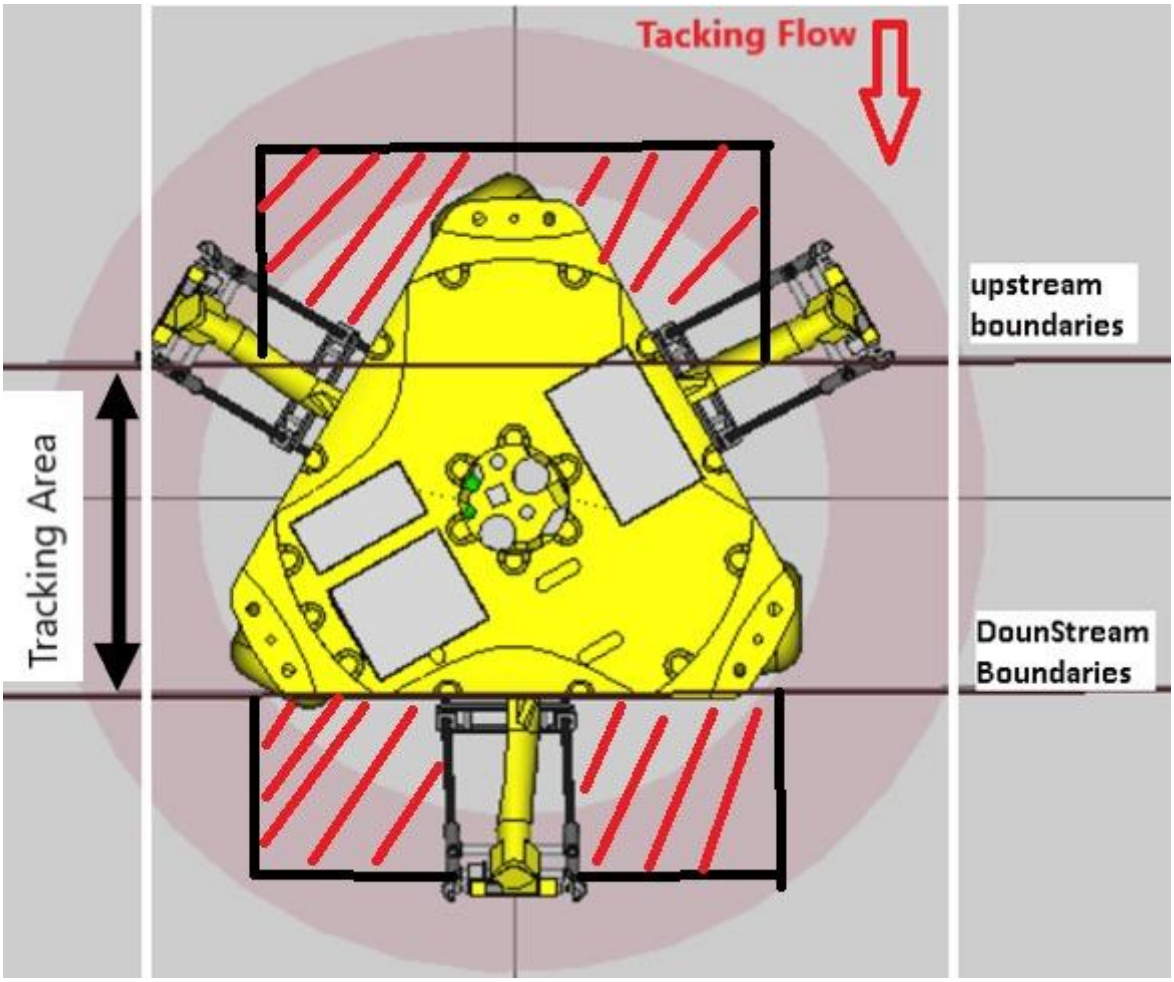


Figure A - Gained areas by applying the dynamic boundaries of upstream and downstream

The implementation increases 30% of cycle speed in terms of pick and place, which consequently increases the whole machine's performance. This development will be implemented in CT Flex robot in the next multi-robot machine, as well as this robot will be the only Delta robot used in such machines.

References

[1] John J. Craig, “Introduction to robotics, Mechanics and Control, Third Edition, 2005.

[2] Clavel, R., Delta, a Fast Robot with Parallel Geometry. 18th International Symposium on Industrial Robot, pp. 91-100, April 1988.

[3] YangminLi, Qingsong Xu,” Dynamic modeling and robust control of a 3 PRC translational parallel kinematic machine”, Robotics and Computer-Integrated Manufacturing journal, Science Direct, 2009.

[4] André Olsson, Modeling and control of a Delta-3 robot, master thesis, Department of Automatic Control, Lund University, 2009.

[5] Angelo Liadis, fuzzy logic control of a two degree of freedom parallel robot, Master thesis, Department of Electrical Engineering, Lakehead University, 2010.

[6] Manuel Napoleon Cardona Gutierrez, "Kinematics Analysis of a Delta Parallel Robot", IEEE, University of Sonsonate, Salvador,2010.

[7] P.J. Zsombor Murray," An Improved Approach to the Kinematics of Clavel's DELTA Robot", Center for Intelligent Machines, McGill University, 2009.

[8] Xia Wu, Jun Lin and Zhencai Zhu," Inverse Kinematics and Singularity Analysis for a 3-DOF Hybrid-driven Cable-suspended Parallel Robot", International Journal of Advanced Robotic Systems, INTECH, 2012.

[9] Mohsen, Mahdi, Mersad," Dynamics and Control of a Novel 3-DoF Spatial Parallel Robot", International Conference on Robotics and Mechatronics,2013.

[10] Serdar Kucuk and Zafer Bingul," Robot Kinematics: Forward and Inverse Kinematics, Industrial Robotics: Theory, Modelling and Control", Sam Cubero (Ed.), ISBN: 3-86611-285-8, InTech,2006.

[11] Reza N. Jazar," Theory of Applied Robotics, Kinematics, Dynamics and control", second edition, Springer, RMIT university,2010.

[12] Yangmin Li and Qingsong Xu," Dynamic Analysis of a Modified DELTA Parallel Robot for Cardiopulmonary Resuscitation", Department of Electromechanical Engineering, Faculty of Science and Technology University of Macau, 2004.

[13] Modeling and High Precision Motion Control of 3 DOF Parallel Delta Robot Manipulator.

[14] Fanuc Robot Series, R-30+B/R-30+B Mate CONTROLLER iRPickTool Operator Manual B-83604EN_04.

[15] Hakan Sahin. Design of a secondary packaging robotic system. PhD thesis, Middle est technical university, Dec 2005.

[16] Gaël Humbert, Minh Tu Pham, Xavier Brun, Mady Guillemot, Didier Noterman. Development of a methodology to improve the performance of multi-robot pick & place applications: From simulation to experimentation. 2016 IEEE ICIT, Mar 2016, Taipei, Taiwan. ff10.1109/ICIT.2016.7475067ff. ffhal01333219f.

[17] Liu, XJ., Wang, J., Oh, KK. et al. A New Approach to the Design of a DELTA Robot with a Desired Workspace. Journal of Intelligent and Robotic Systems 39, 209–225 (2004).

[18] T. Fochi, Dynamics of Delta Robot, Rockwell Automation, 2009.