POLITECNICO DI TORINO

Department of Electronics and Telecommunications

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

End-effector tools wear prediction: interaction with the workpiece modelling in a quasi-FEM approach

Candidate:
Dario CANIGLIA

Mentor:	Prof. Alessandro RIZZO
Supervisor:	Ing. Giovanni GUIDA - Brain Technologies

December 2021

Abstract

Nowadays, in the "Industry 4.0" context, an important topic is the State of Health (SOH) estimation of machines because, knowing the SOH of machines, it is possible to intervene before that some damage could occur. Thus, the companies can increase the performance of their production systems and to reduce the maintenance costs. The MorePRO project, owned by Brain Technologies, fits in this context. This project has the goal of developing a real-time system able to estimate the SOH of the end-effector tools in a CNC machine. The technique that Brain Technologies wants to develop is able to estimate the friction coefficient to which the tool is exposed during metal cutting operations, extracting from this parameter the information to estimate the SOH of the end-effector tool.

This thesis goal is to start the development of a similar FEM (Finite Element Method) analysis in Mathworks environment to obtain a model of the tool interaction with the work piece. From this model, artificial data can be obtained and they can be used to develop estimation methodologies. In this thesis work, my role was the development and implementation of an algorithm to detect the part of the end-effector tool that is employed in the metal cutting operation. The main activities performed can be divided as follow:

1. **Creation of a model of the CNC machine**: the CNC machine analysed was considered divided in two parts. The first part has the objective to place the end-effector and the second part to place and orientate the work piece. Models of the two parts were created using the acknowledgement acquired in the Robotics course. These models were necessary to obtain the 3D coordinates of the end-effector tool and the work piece.

2. **Creation of trajectories in the operational space**: to simulate the CNC machine movement, it was necessary to give to our model a trajectory as input. This analysis is interested in the movements of the end-effector tool; thus, the trajectories were developed in the operational space. A segment in the space trajectory and a circumference in the space trajectory were developed.

3. **Identification of the interaction points**: the end-effector tool was approximated to a point cloud. The points that interact with the work piece during the movement were identified and they will be used in a mechanical analysis.

Acknowledgements

First of all, I would like to express my acknowledgements to Prof. Alessandro Rizzo and to Ing. Giovanni Guida for the guidance and encouragement they have given me throughout this thesis work.

I would like also to thank my course mate Emanuele Pansica with whom I worked on this project. Without his help and encouragement this thesis work would have been more complex.

A special dedication to my parents and friends that have supported and encouraged me during my studies.

Table of Contents

List of Tables

List of Figures

Acronyms

FEM

Finite Element Method

SOH

State of Health

CNC

Computerized Numerical Control

Chapter 1

Introduction

In the last few decades, technology has experienced an incredible development. The new technologies created were employed by companies in the industrial automation sector to improve working conditions, create new business models and increase the productivity and the products quality. This integration of these new technologies is known as "Industry 4.0". The most important principles of the "Industry 4.0" are:

- Interconnection: different devices are able to communicate via the Internet of things;
- Information transparency: operators can obtain complete information about the manufacturing process, using it to improve the process;
- Technical assistance: systems are able to assist the operator with unsafe task or in problem-solving;
- Decentralized decision: systems can take decisions and execute their tasks autonomously.

The MOREPRO project fits in the context of "Industry 4.0".

1.1 Wear estimation

The real-time wear estimation problem is very important in the context of "Industry 4.0" because by knowing State of Health (SOH) of the production machine it is possible to intervene with a predictive maintenance in advance to avoid a failure. Thus, the production efficiency and safety can be increased, and the costs can be reduced. For these reasons, in the last few years the real-time estimation of the SOH of the production machine and the predictive maintenance techniques have become a relevant topic in scientific research.

The predictive maintenance has a long history. Its first form was the visual inspection but, with the advancement of technology, it evolved to automated methods that can employ different techniques as neural networks, fuzzy logic, and data-driven empirical and physical modelling. This evolution was feasible thanks to the sensors that can be integrated in the machine systems, from which it is possible to obtain some important parameters to use in the State of Health prediction [1]. The main tasks of these predictive maintenance technique are:

- State of health estimation of the machine different components;
- Computation of solution to prevent a failure.

The importance of predictive maintenance techniques that use predictive algorithms is explicable because these methods permit to identify a critical situation before it occurs and, thus, to intervene with reparations or the substitution of the machine. This is very useful in high precision processing because, if a failure occurs during the working process, it would have consequences on the workpiece. Thus, the company would suffer an economic damage because of the workpiece injury and because of a failure that could have been prevented with a maintenance operation in the correct moment. Unfortunately, the SOH estimation is complex and often also expensive. For these reasons, these topics have become very popular in the research environment.

1.2 State of Health of a CNC machine

CNC (Computer Numerical Control) machines are usually employed in machine tool processes as metalworking. They process a workpiece to obtain the desired shape in accord to programmed instructions. The desired shape is usually very complex and it can not be obtained with a manual control of the machine, for this reason, it is used a machine controlled by a computerized system. Consequently, the precision is guaranteed and the efficiency is increased. The work process consists of the remotion of material parts from the workpiece using an end-effector tool. Thus, the most important part of a CNC machine is its end-effector tool because it is in contact with the material and its failure would produce a damage on the workpiece. Thus, the SOH estimation of the end-effector tools has a notable importance. Unluckily, the modelling of the interaction between the tool and the workpiece is very difficult because there are numerous variables to consider. The most significant are:

- Configuration of the robot;
- Environmental parameters;

- State of Health of the tool.

Furthermore, it is not possible to directly estimate the wear condition of the tool. Sensors can be used to measure some relevant parameters as temperature, voltage, forces, pressure and from these measured values a rough value for the SOH can be obtained. However, this estimate is not accurate enough because sensors are usually not set up in the right position, to not interfere with the working process. For example, a temperature sensor cannot be positioned too close to the end-effector tool, thus the value received from the sensor is not accurate and it is subject to errors. Moreover, the measured parameters could not be sufficient to estimate the real condition of the end-effector tool.
Currently, the most used methods in the State of Health estimation are machine learning technique combined with digital-twin simulation. The disadvantage of these methods is the high computational requirements because of the numerous parameters that are presented in the cutting process. Between these parameters the most important are the mechanical ones because they can provide information about the stress to which the end-effector tool is subjected. From the mechanical point of view, the principal parameters to consider are:

- Friction coefficient;
- Temperature;
- Chip load.

The contact forces involved in the working process influence greatly the parameters previously mentioned.

1.3 MOREPRO project

As it is intuitable from the previous arguments, the aim of MOREPRO project is the SOH estimation of a CNC machine. The project is based on a logical architecture distributed in three levels:

- To monitor the SOH of the critical components of the machine. By means of some embedded sensors data are acquired and, successively, they are elaborated with machine learning and data mining techniques, to have information about the SOH of the machine.
- To monitor the end-effector tool wear using digital twin tools. Comparing real-time environment signals with some estimated quantity in a specific simulation environment, some useful information about the end-effector usury can be extrapolated.

- To predict the machine SOH evolution in time, in order to implement a preventive maintenance, avoiding that the production process could be stopped.

The architecture is divided in two layers. The first level is composed by edge devices which acquires information from the system and, elaborating it, they provide a real-time supervision of the system's SOH. Thanks to this first level, it is possible to intervene immediately when a danger is detected, avoiding the occurrence of a failure. Successively, the data acquired from the system are sent to the second layer which is a server on the cloud. A large amount of data is collected and elaborated employing digital twin techniques in order to compute some parameters to reconfigure the edges device. With this parameters update, it is ensured that the edge devices performance is always optimal and their behaviour can be adapted to the current system situation. The figure 1.1 illustrates a general scheme of the MOREPRO project.

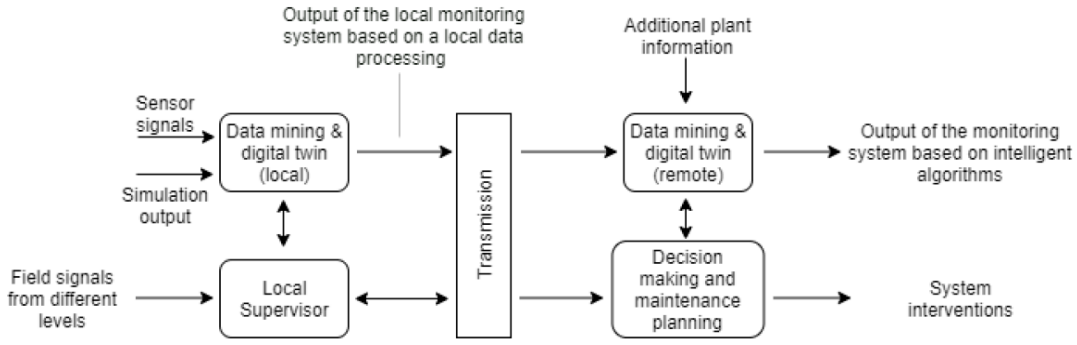

Figure 1.1: General structure of MOREPRO system

The *data mining and digital twin (local)* in the previous figure will be implemented in the edge devices. They will acquire data from the sensors and a simulation and will exchange information with the *local supervisor*. Then, the edge devices will elaborate these data to compute the SOH value, in order to monitor the machine. The *local supervisor* reconfigures the edge devices parameters on the base of the information provided by the server on the cloud.
Usually, the wear estimation techniques are based only on deep machine learning calculations, requiring that this operation is performed on the cloud because of the large amount of data and the processing complexity. However, in the MOREPRO architecture, the use of edge devices permits to elaborate data locally, as much close as possible to the source. This solution permits to increase the performance obtaining a real-time monitoring with low costs. Moreover, the problems which could arise from the internet connection are avoided. Lastly, it is very simple to

modify their code or the architecture in which they are employed. For these reasons, the MOREPRO architecture is very innovative.

1.4 Partnership

In the MOREPRO project different companies are involved. Each company is specialized in a specific field and recover an important role to achieve the purpose of this project. The list of these companies is exposed below:

- Brain Technologies: it is the MOREPRO project manager. It contributes to the definition and design of the digital architecture. Furthermore, it is working on the software development of the predictive system both on the edge devices and on the cloud. To conclude, it is responsible of the implementation, testing and validation of the final prototype.
- MCM S.p.A: it defined the algorithm's requirements and constraints based on the user needs. Furthermore, it is responsible to interface the machine for data collection, which also provides the installation of new sensors, and to support the integration of the MOREPRO solutions with the CAMS machines. Lastly, it will contribute to the testing and validation phases of the prototype.
- AL.MEC: it is a company of mechatronic components. Its task is the design and production of specific electronic boards, which act to acquire data from the sensors and to elaborate them.
- CAMS: it is a company for high precision machining. Being an expert of this sector, it shared its vision and expertise. In particular, it contributes to the definition of the algorithm's requirements and with the analysis of the cases of interest in industrial application. Finally, the implementation, testing and validation phases will be performed on its production line.

1.5 Thesis outline

The purpose of the project's part on which I worked is the development of an accurate model of the CNC machine able to produce artificial data useful to develop suitable algorithms to estimate the state of health. I worked on this task with Emanuele Pansica, a course mate. We made together the kinematic model of the CNC machine to simulate the machine movements and the trajectory planning part. Then, I focused my attention on the interaction between the end-effector tool and the workpiece, in order to identify the contact points during the simulation of the cutting operation. Then, the obtained contact points will be used in a mechanical

analysis to whom my colleague Emanuele is working on. Below, it is exposed an overview about the contents of the chapters of this thesis.
In chapter 1, a brief introduction about the wear estimation problem and its complexity was presented. Then, the MOREPRO project was described, highlighting the general architecture developed by Brain Technologies and its partners, explaining the advantages of this method. Lastly, all the companies involved in this project were presented.
Chapter 2 contains an overview about the different methods to estimate the SOH present in literature, highlighting their advantages and disadvantages. Successively, the work carried out by the previous MOREPRO team has been generically analysed. Finally, the Finite Element Method (FEM) was briefly exposed.
In chapter 3 was described generically the CNC machines and, in particular, the one analysed. Then, after the description of the Denavit-Hartenberg convention, it was illustrated the kinematic model of the CNC part which moves the end-effector tool and the algorithm used for the kinematic inversion. Lastly, a similar analysis was applied to the second part of CNC machine which positions and orientates the workpiece.
In chapter 4 was illustrated the trajectory planning part. The theory on the base of this task was explained, highlighting the reasons behind our choices. Afterwards, it was explained how we computed a linear trajectory and a circular one to use as input for the CNC model described in the previous part.
In chapter 5, it was analysed the process used to approximate the end-effector tool to a point cloud and to identify the contact points with the workpiece during the simulation of the cutting process.
Lastly, the conclusions were presented, and a brief analysis of the future works was outlined.

Chapter 2

State of the Art

For the modern companies, as said in the previous chapter, it is extremely important to improve the working process efficiency and to reduce costs. For these reasons, the develop of advanced system maintenance technique is extremely important. The standard EN 13306 (2001) (figure 2.1) divides the maintenance techniques in two categories:

- **Corrective maintenance**;
- **Preventive maintenance**.

The corrective maintenance is defined as "maintenance carried out after fault recognition and intended to put an item into a state in which it can perform a required function" [EN 01]. Thus, this technique is used only when a possible failure cannot have serious consequences and the repairs are not expensive and they can be performed in a short time. This category of maintenance can be further divided into palliative maintenance and curative maintenance. In the first, the repair is temporary; in the second, it is definitive. The preventive maintenance is defined as "maintenance carried out at predetermined intervals or according to prescribed criteria and intended to reduce the probability of failure or the degradation of the operation of an item". This type of maintenance is divided into predetermined maintenance and condition-based maintenance. In the first, the components of the production machine are replaced at fixed time intervals, without a check to verify if it is sufficiently degraded or not. Usually, this technique leads to a waste of resources. In the second, a possible failure can be identified by the real-time analysis of data acquired from the system. It has the goal of discovering anomalies in the working process and to act consequently. The problem is that this technique does not provide certain results. To compensate this uncertainty, the predictive maintenance was introduced. It tries to predict the system State of Health evolution using the current system condition [2].

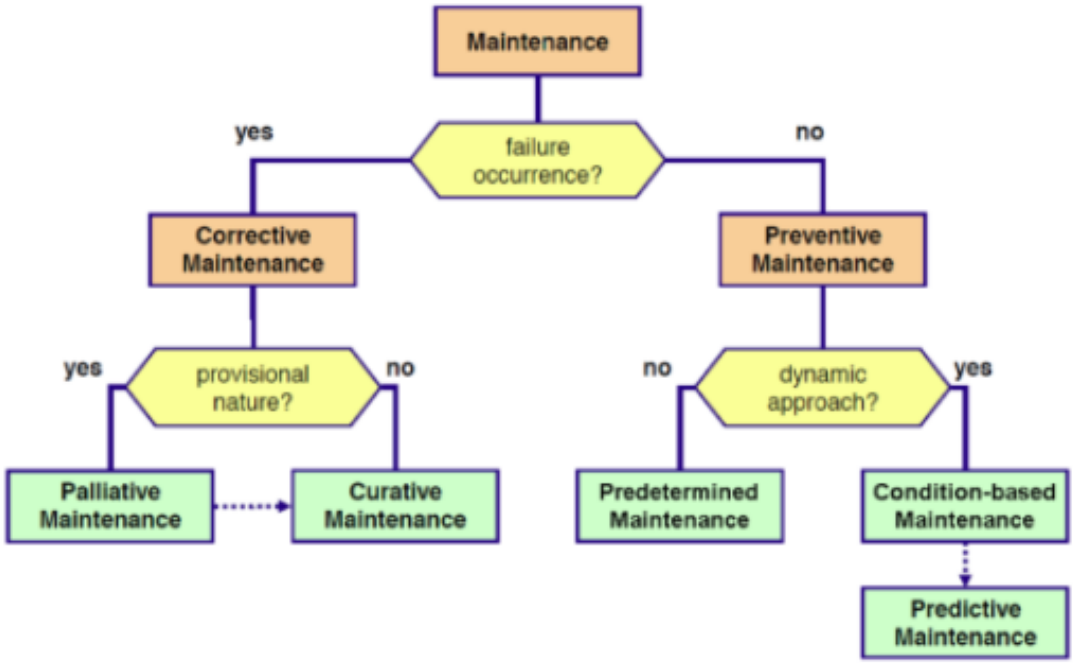

Figure 2.1: Forms of maintenance according to the standard EN 13306 (2001)

2.1 State of Health estimation methods

The predictive maintenance is complex topic, to solve the State of Health estimation problem different technique were employed. The model on which I worked will produce artificial data used to develop a suitable method for estimating the SOH. Different methods could be employed. Some of them are exposed in the following part.

2.1.1 Model-based fault detection

This technique is based on a mathematical model of the process, used to predict the State of Health evolution in time. This is possible considering a system model that describes both its normal and faulty behaviour. Afterwards, some significant system parameters are measured and compared with the model outputs. On one hand, when the machine works in normal conditions, the results of this comparison have small values; on the other hand, when a failure is approaching, this comparison produce significant values. These data are used to develop a statistic representation to estimate the residual lifetime of the machine. The models used in this approach can be divided into two categories [3]:

- **Physics-based models**: they are obtained by using physics laws to describe the behaviour of the system macroscopic parameters. These kinds of models produce an optimal result when the modelling process is very accurate. For this reason, it is important to also analyse the microstructural material characteristics. The disadvantage of this kind of models is the high complexity.

- **Parameter estimation models**: this approach is used when it is not possible to compute an adequate physics-based model because the system is too complex. This method consists in the use of identification procedures. Initially, considering some information about the system, some hypotheses are made about the model structure and complexity. Then, the coefficients of a mathematical function that represents the system are computed. The disadvantage of these models is that there is not a precise relation between the physical parameters of the system.

2.1.2 State observer

The use of state observer is very common in State of Health estimation. There are different kinds of state observer, each one is appropriate to a different case. The Kalman Filter is used when the system has a linear behaviour with additive Gaussian noise. Instead, with a nonlinear behaviour it is employed an Extended Kalman Filter. When the system is nonlinear and the noise is non-Gaussian, the Particle filter and the Bayesian theory are used [3].

- **Kalman Filter**: it is an algorithm that estimates the state using a system's model and the output measurement. The advantage of this method is the capability of computing the state estimate using only measurements from the current time instant. Thus, it is not required to store the past measures. The Kalman Filter algorithm structure is represented in figure 2.2. The Kalman Filter algorithm is composed by two different steps: prediction and innovation. In the first phase, a predicted state estimate $\widehat{x}_{k|k-1}$ is computed using the state estimate of the previous time instant. In the second phase, the state estimate $\widehat{x}_{k|k}$ is calculated combining the predicted state estimate with the current observation information.

- **Extended Kalman Filter**: it is used with nonlinear systems with additive Gaussian noise. The system is linearized and, thus, it is treated as a linear one. The algorithm is the similar to the Kalman Filter, with two steps: prediction and innovation.

- **Particle filters**: they are also known as Sequential Monte Carlo methods. They are used with nonlinear systems with non-Gaussian noise. They approximate the posterior state distribution, combining the state estimate and its

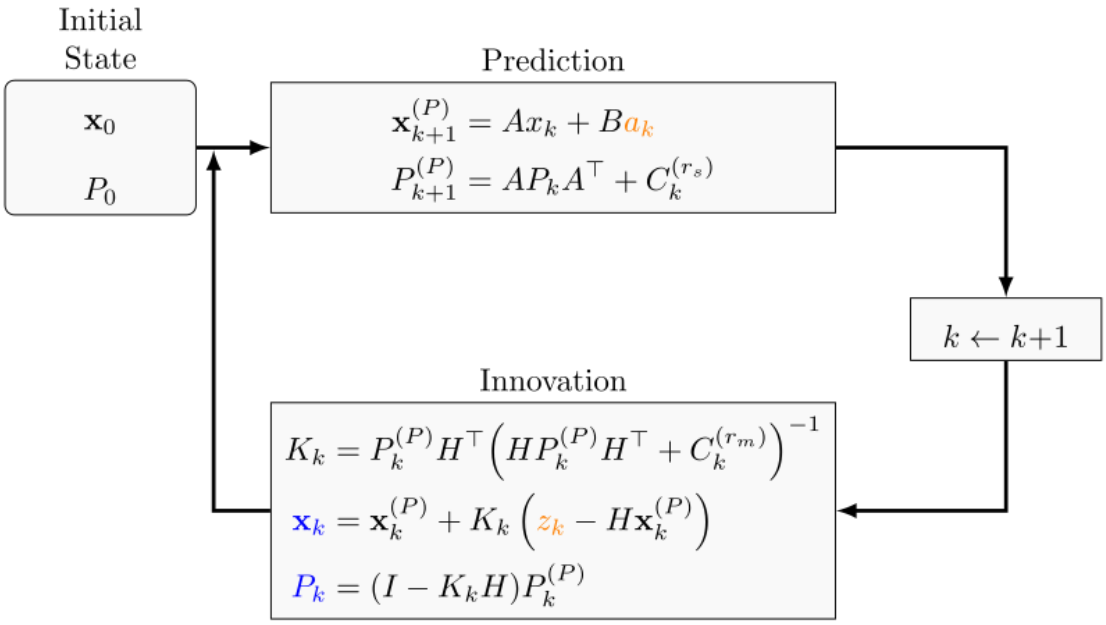

Figure 2.2: Generic structure of Kalman Filter algorithm

associated uncertainty to generate weighted samples, called particles. With a higher number of particles, these filters have better performance. Considering the State of Health estimation, these filters act in two phases. In the first, the current fault dimensions and the changing parameters in the environment are estimated. In the second, the data obtained in the first operation are used with the fault growth model to generate state prediction from (τ+1) to (τ+p). Then, from this prediction, the prognosis confidence interval is estimated.

2.1.3 Data-driven fault prognosis

These techniques use measurement signals and their statistics to create nonlinear structure. These methods are easier and cheaper than the model-based ones. However, they require numerous training data and they have a potentially wide confidence interval. This category includes numerous methods, the most relevant are:

- **Artificial neural networks**: their structure (figure 2.3) imitates the biological nervous system. Using numerous data as input for the network, some training algorithms map a relation between the input and output of the system. The network is composed by numerous structures called neurons. The weight of the neurons is adjusted to obtain a better approximation of the system;

thus, the artificial neural networks have a self-adaptive structure. After this training procedure, the neurons weights are set and the network can generate the desired output, in this case, the fault evolution prediction [3].

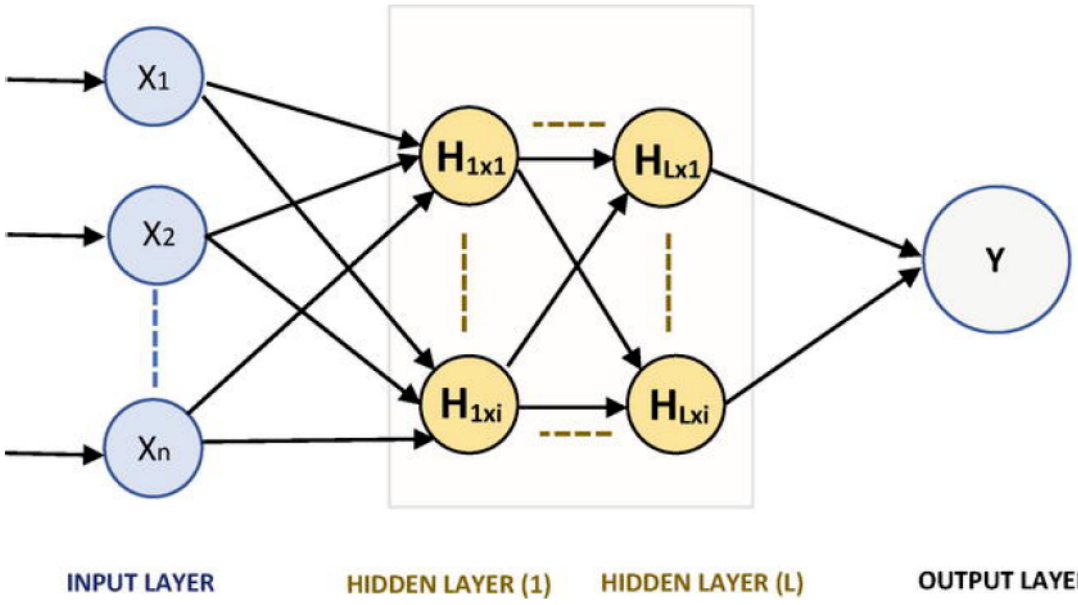

Figure 2.3: Structure of the artificial neural network

- **Fuzzy-logic**: it is similar to the previous one, it also provide mapping between input and output. They imitate the linguistic and reasoning human capabilities. They are based on if-then rules and, adjusting membership functions, they provide useful data to predict the system condition. Usually, a fuzzy-logic

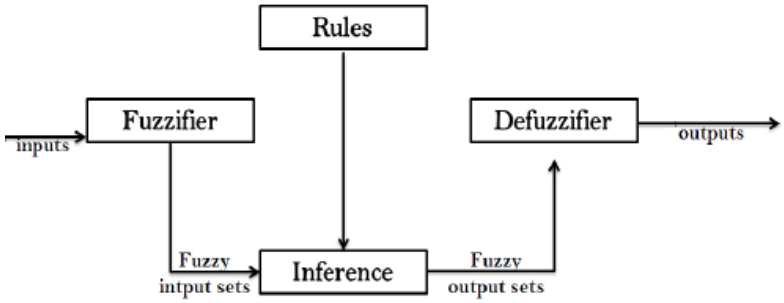

Figure 2.4: Components of a fuzzy-logic system

system has three different phases (figure 2.4): fuzzification, inference and defuzzification. In the first, a set of input data is converted to a fuzzy set using fuzzy linguistic variables. In the second, an inference is made based on a set of rules. In the last, the output is mapped using the membership functions. In predictive maintenance, the final result is a curve that provide useful information about the wear conditions [4].

- **Hidden Markov Models**: it is a statistical model used to describe system transition between states. It is the extension of the Markov chain when the system present unobservable or partially observable states. Figure 2.5 shows the general structure of a discrete-time hidden Markov model with N states, S=($s_1, s_2, \ldots, s_N$) and M observation symbol, V=($v_1, v_2, \ldots, v_M$). The states are interconnected, thus, a transition between any two states is possible. Considering the state q_t at time t, it depends only on the state q_{t-1}. The result of hidden Markov models is λ=(A,B,π), where the transition matrix A contains the probability that the state j follows state i; the observation matrix B contains the probability that observation k is produced from the j-th state; the initial state array π contains information about initial probabilities. Using the Baum-Welch training algorithm, the hidden Markov model can be used to obtain an estimation of the system health. Moreover, it can provide the probability that the system is in a specified state after n iterations [3].

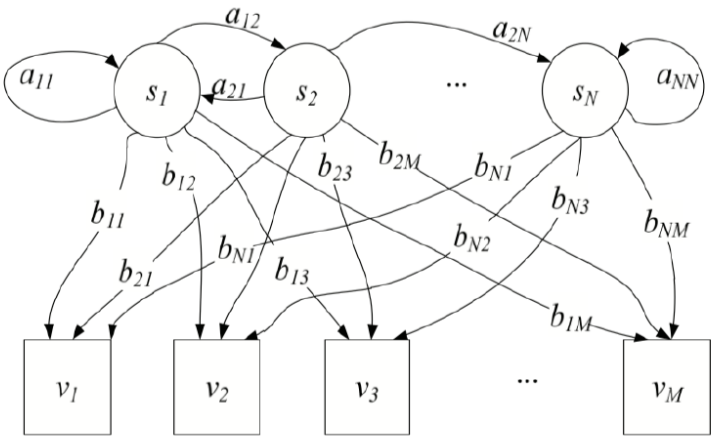

Figure 2.5: General structure of discrete-time hidden Markov model

2.1.4 Vibration monitoring

This technique, as said the name, estimate the State of Health by using the vibrations as indicator. Vibrations contain mechanical information about the system, in this way, it is possible to detect a possible damage. The disadvantage of this method is that it is more useful in diagnostic aspects than prediction ones. There are, however, some application in the preventive maintenance topic. Looking at the PF-curve (figure 2.6), it represents the machine 's condition before it reached the failure. There are two different regions of the curve can be identified. In the first one, on the left, the condition changes slowly until at a certain time instant the point of observability of deterioration (P) is reached. The second part is used to predict roughly the behaviour of the system's State of Health [5].

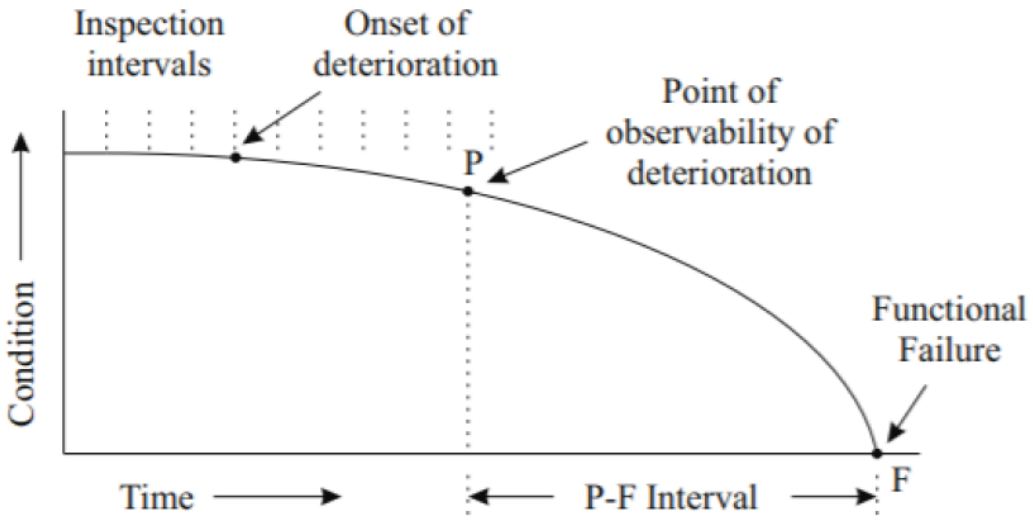

Figure 2.6: P-F curve

2.1.5 Mixed/Hybrid algorithms

The different techniques previously described possess advantages and disadvantages, for this reason, some methods which combine some of them were developed. In this way, the disadvantages are compensated, and the quality of the State of Health estimation is increased.

2.2 Results from past MOREPRO team

The MOREPRO project was already underway when my thesis work started. Thus, as first step, the work of past MOREPRO team was analysed. The team was dived in three sub-teams, each of them with a different task [6] [7] [8]:

- Prediction team: : using a simple model, they tested an approach based on extended Kalman filters and error analysis techniques on MATLAB and Simulink environment, in order to estimate wear and SOH of a CNC machine.
- Modelling team: they developed a preliminary kinematic and dynamic model of the CNC machine.
- Requirements team: they analysed requirement and specifics for the different parts of the project, and they developed a design of experiment.

For my work purpose, the most interesting part was the modelling one. It was a starting point for my analysis because they tried to design a simple model to describe the interaction between the end-effector tool and the workpiece. This is the crucial part of the project phase on which I worked, in fact, from the study of this interaction the artificial data will originate.
The past team considered initially a simple model of the end-effector tool. It consists of a rotational disc that translates in the workpiece direction to cut it (figure 2.7). They started with a simple mechanical analysis, considering the forces and torques balance to obtain the model of the mechanical part. Afterwards, they considered the motor which moves the mechanical part of the CNC machine. Usually, modern CNC machines are driven by a brushless or servo motors. However, to simplify the analysis, a DC motor was considered, and they produced a model of the electric part. Finally, combining the two different models, they obtained the following model:

$$\begin{cases} \ddot{\theta} = \dfrac{k_t i_a}{I_n} - \dfrac{Att_{mot}\dot{\theta}}{I_n} - \dfrac{\beta F_c \dot{\theta}}{I_n} \\ \ddot{x} = \dfrac{F_1}{m} - \dfrac{F_c(F_2\alpha + c)}{m} \\ \dot{i}_a = \dfrac{V_a}{L} - \dfrac{Ri_a}{L} - \dfrac{k_v\dot{\theta}}{L} \end{cases}$$

where the states are:

- $\dot{\theta}$: Rotational velocity;
- $\dot{x}$: Linear velocity;
- i_a: DC current of the motor.

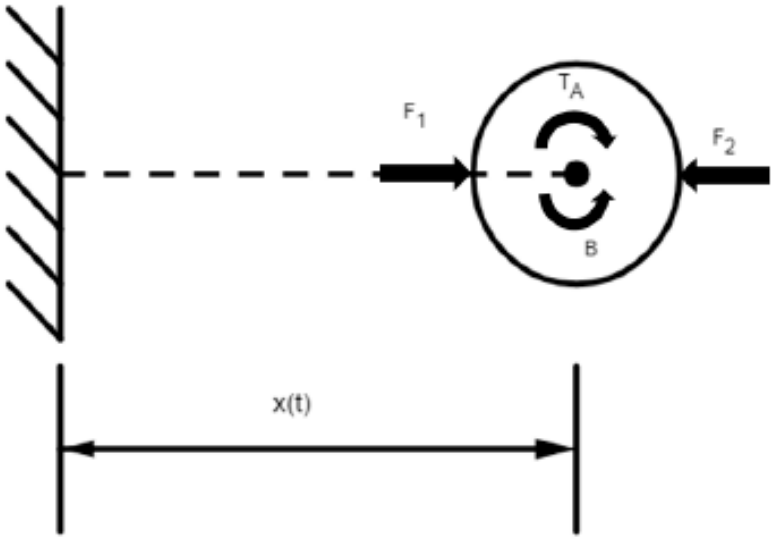

Figure 2.7: Simplified milling machine model

The model input are:

- V_a: armature voltage;
- F_1: horizontal force that moves the cutter;
- F_c: function that defines the contact with the workpiece. Its value is 1 when there is the contact and 0 in the other case.

The model parameters are:

- β: friction coefficient;
- L: inductance of the motor;
- m: tool's mass.

As already said, this model is very simple, thus the previous team tried to improve it adding some additional information about the interaction between end-effector tool and workpiece. In particular, they looked for some model in literature about the friction coefficient, in order to perform the integration in the model in development. They chose the friction coefficient because it is an important parameter in the cutting operation, and it can provide important information about the SOH. They found numerous different analyses for the friction coefficient because the cutting

process is very complex and the friction is influenced by numerous parameters such as the temperature, the materials, the cutting velocity, the applied forces, the contact surface. Between the different analyses examined in the previous phase, two of them contributed to produce the idea behind the model on which I worked: cutting force analysis and thermal analysis.

2.2.1 Cutting force analysis

This analysis considers the cutting operation from the mechanical point of view. It provides a relation between the friction and the forces applied in the work process. The cutting forces are estimated considering the cutting tool divided into small elements and applying a mechanical analysis to them. Considering the interaction, the cutting force is decomposed into dF_t, dF_r and dF_a, the three elements along tangential, radial and axial direction (figure 2.8).

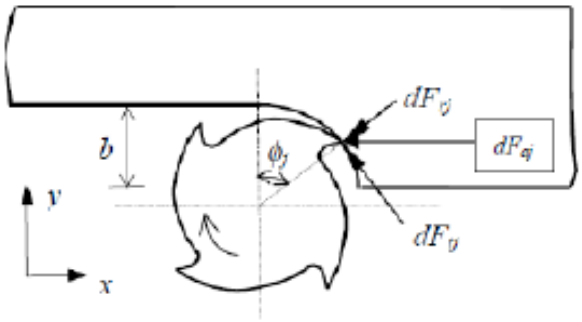

Figure 2.8: Cutting force decomposition in the milling process

These elements are computed with the following equations [9]:

$$\begin{cases} dF_t(\phi, z) = K_t h_j(\phi, z) dz \\ dF_r(\phi, z) = K_r dF_t(\phi, z) = K_t K_r h_j(\phi, z) dz \\ dF_a(\phi, z) = K_a dF_t(\phi, z) = K_a K_t h_j(\phi, z) dz \end{cases}$$

Where h_j is the chip thickness in point j, K_r, K_t and K_a are the cutting force coefficients and ϕ is the shear angle. Thus, these forces are used to compute the components of cutting force along x, y and z direction:

$$\begin{cases} dF_{xj}(\phi) = dF_{tj} \cos(\phi_j) + dF_{rj} \sin(\phi_j) \\ dF_{yj}(\phi) = dF_{tj} \sin(\phi_j) - dF_{rj} \cos(\phi_j) \\ dF_{zj}(\phi) = dF_{aj} \end{cases}$$

Finally, considering these forces and the initial rake angle γ, the friction coefficient β is computed as:

$$\beta = \tan\left[\arctan\left(\frac{dF_{yj}(\phi)}{dF_{xj}(\phi)}\right) + \gamma\right]$$

2.2.2 Thermal analysis

The friction is influenced by the temperature reached during the cutting operation. However, it is complicated to find a suitable relation between these two parameters because the temperature is not constant in the cutting area. It has its maximum value in the tool rake area, near the cutting edge, and the more the distance from this area increases the more its value decreases (figure 2.9). Moreover, it was observed that the temperature is strongly influenced by the cutting speed [10].

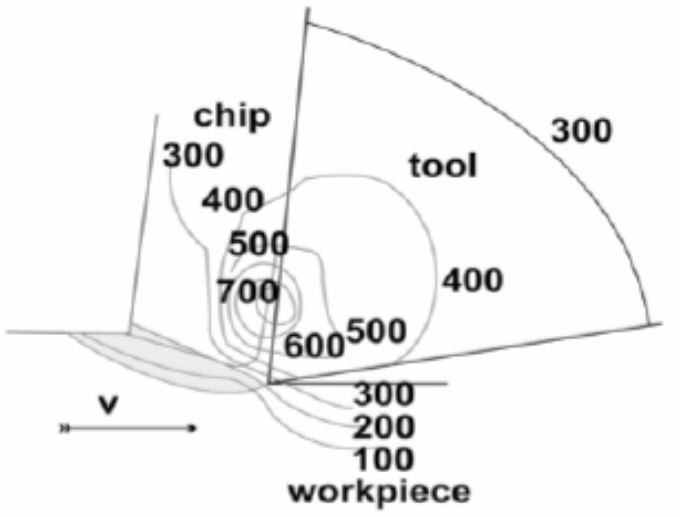

Figure 2.9: Temperature distribution in the contact area

2.3 Finite Element Method

From the previous analyses, we noted that it could be useful to study the system from a microscopical point of view. Thus, we had the idea of decomposing the end-effector tool into small elements to generate a model of the cutting process. Firstly, these elements can be used for a mechanical analysis of the process but successively some other information could be introduced, as the temperature distribution on the contact surface, in order to obtain a more precise model. Consequently, we

decided to take inspiration from the Finite Element Method (FEM) to achieve our purpose.
The FEM is a numerical method to solve differential equations in modelling problems. The idea behind this method is to divide a system into smaller parts called finite elements and each element is considered with homogeneous characteristics. Thus, it is possible to define a set of equation for each of them and, afterwards, recombining the element equations to obtain the final solution. The advantage we are most interested in is that this method permits to capture the local effects.
The FEM was already used to analyse different phenomena in the CNC machine cutting process [11] [12]:

- material removal;
- computational models for specific machining processes;
- thermal aspects in machining;
- residual stresses in machining;
- dynamics analysis and control of machine tools;
- tools, wear and failure;
- chip formation mechanism.

This type of analyses produced good results, consequently, we decided to develop our model on the base of this method. Initially, considering only the mechanical behaviour of each finite elements and successively adding other details to improve the model. The first step in this direction is to model the CNC as a robot to identify the position of the end-effector tool during the work process. Secondly, to decompose the tool in finite elements and to identify the elements that interact with the workpiece in each time instant. These elements will be used in the mechanical analysis to generate the artificial data.

Chapter 3

CNC machine modelling

The goal of this thesis work, as already said, is to develop a CNC simulator able to produce artificial data about the cutting process. To achieve this purpose, the first step is to create a CNC model able to reproduce the movements of the machine. In this chapter, after an overview about CNC machines and, in particular, about the one analysed, it is shown the process applied to obtain this kind of model.

3.1 CNC machine

The CNC (Computer Numerical Control) machine, as already said, is an automated high precision machine. A computer manages the process operation and it provides information about speeds and the position of the cutting tool. The CNC machines have typically a SCARA or a cartesian robotic configuration (figure 3.1).

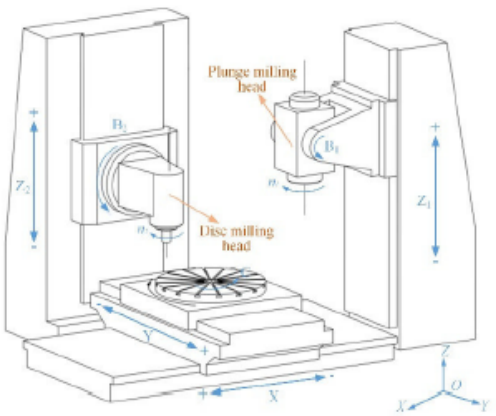

Figure 3.1: Example of CNC machine structure

Some of the advantages of CNC machines are the following:

- The high accuracy and precision permit to execute tasks which are impossible for manual machines;
- The part produced have the same accuracy, without variations between them;
- It is possible to produce a complex design in the minimum time possible;
- Reduce the labour costs because it is necessary a small number of operators.

CNC machines are used in machine tool processes which consists in removing material from a workpiece to obtain the desired shape. The cutting tool can interact with the workpiece in different modes: turning, drilling, milling and grinding, shown in figure 3.2. For each of these operations, it is necessary to use a different tool and to provide different cutting parameters: cutting speed, depth of cut, feed [13].

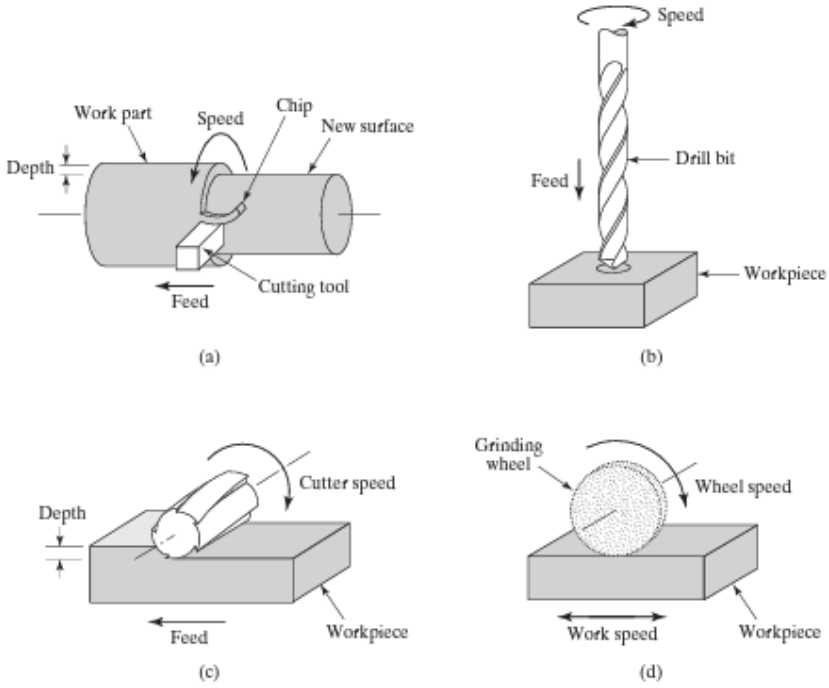

Figure 3.2: The four common machining operations: (a) turning, (b) drilling, (c) peripheral milling and (d) surface grinding

The cutting speed is the relative velocity between the work surface and the tool, it is measured in m/min. Usually, to the CNC machine computer is provided the cutting speed converted into spindle rotation speed, measured in rev/min. The

feed is the size of the chip formed by each tooth in the work process, mm/tooth. Usually, it is provided to the computer as feed rate, mm/min. The depth of cut is the distance the tool penetrates inside the workpiece, mm.
The CNC machine analysed in my thesis work is shown in figure 3.3. It was considered as composed of two different parts. The first part is composed of two prismatic joints and it positions the end-effector tool, moving it along the x and y direction. The second part moves and orientate the workpiece by means of a prismatic joint and two revolute joints. To simplify the kinematic analysis, these parts were considered and examined independently. It was assumed that they work separately: the second part place the workpiece in the correct position and, only after that, the first part moves the end-effector to start the cutting process.

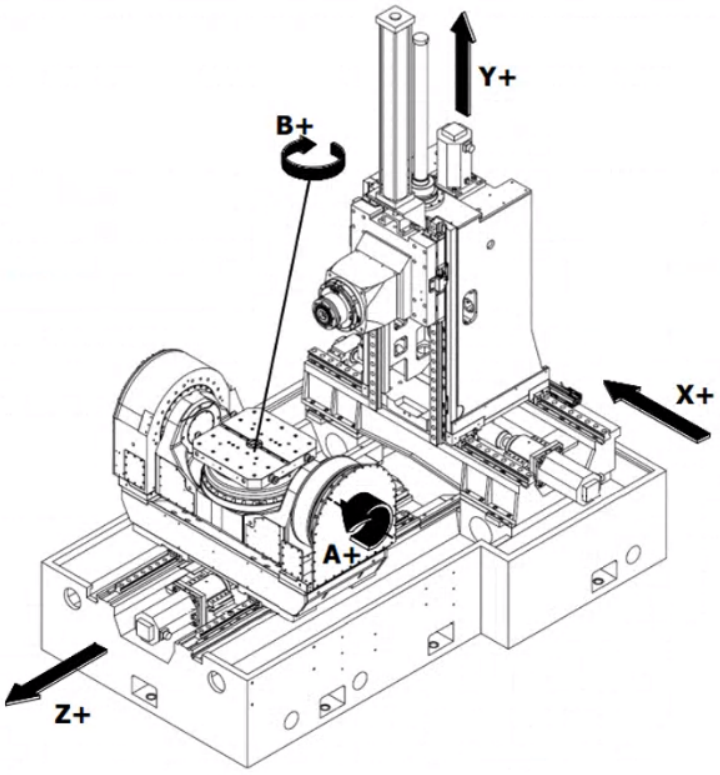

Figure 3.3: Structure of the analysed CNC machine

3.2 Direct Kinematics

In this section it is explained the direct kinematics analysis of the CNC machine. This is the first step to produce the part of the CNC machine simulator that allows

the end-effector movement.
A manipulator can be schematically represented from a mechanical point of view as a kinematic chain of rigid links connected by means of joints. This representation is useful to derive a relation to compute the end-effector pose. There are two types of joints:

- **Prismatic**: it produces a linear movement between links;
- **Revolute**: it produces a rotation along an axis.

To obtain the end-effector pose is used a relation between the joint's coordinates. The method employed to find this relation is based on the Denavit-Hartenberg convention. Firstly, this method will be explained and then it will be applied to our case.

3.2.1 Denavit-Hartenberg convention

Denavit-Hartenberg convention is a systematic method to compute the direct kinematics equation for an open-chain manipulator. It defines the relative pose of two consecutive links as a homogeneous transformation matrix and, then, all the matrices are used to compute the end-effector pose. The first step is to define the reference frames attached to each link. A reference frame is attached to a link when each point of the generic link i has constant coordinates with respect to frame i, this means that when link i is actuated, frame i moves with it. The only frame that does not move is the frame 0 associated with the base of the manipulator, it is fixed.
For a manipulator composed by n joints and $n+1$ links, it is necessary to define $n+1$ frames. The Denavit-Hartenberg convention fixes a precis methodology to define these frames. Considering the generic joint i that connects link $i-1$ and i in figure 3.4, the Denavit-Hartenberg convention is applied to define link frame i [14]:

- Axis z_i is positioned along the axis of joint $i+1$;
- Considering the common normal to axes z_{i-1} and z_i, the origin of frame i O_i is positioned on the intersection between axis z_i and the common normal. Moreover, the origin $O_{i'}$ is positioned at the intersection between the common normal and axis z_{i-1}.
- The axis x_i is chosen along the common normal identified in the previous step, the direction goes from joint i to joint $i+1$.
- Axis y_i is chosen to obtain a right-handed frame.

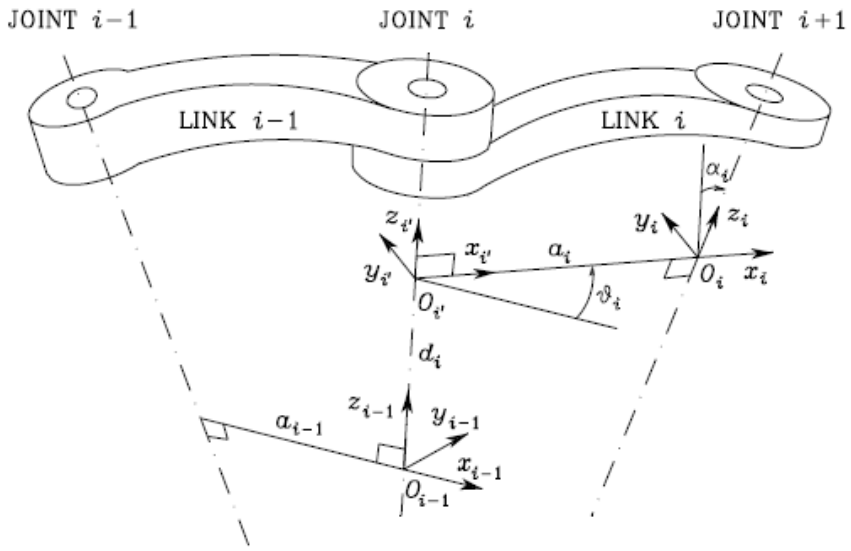

Figure 3.4: Example of a Denavit-Hartenberg application

From the previous rules, it is possible to identify some unique cases:

- For frame *0*, only the direction of axis z_0 is specified. The O_0 and x_0 can be chosen arbitrarily;
- For frame n, z_n is not uniquely defined. Typically, it is chosen aligned to z_{n-1};
- When two consecutive axes are parallel, axis x_i is not uniquely defined;
- When two consecutive axes intersect, the direction of x_i is arbitrary;
- When joint i is prismatic, the direction of z_{i-1} is arbitrary.

After the definition of all the reference frame, it is necessary to identify four parameters used to compute the transformation matrix from frame i to frame *i-1*. These parameters are the following:

- a_i: the distance between O_i and $O_{i'}$;
- d_i: the coordinate of $O_{i'}$ on axis z_{i-1};
- α_i: the angle between axes z_{i-1} and z_i about axis x_i;
- θ_i: the angle between axes x_{i-1} and x_i about axis z_{i-1}.

The parameters a_i and α_i are always two constants and depend on the geometry of the considered part. Among the other two, one is a constant and the other is a variable, it is the type of joint that establish which is the variable. When the joint is prismatic the variable is d_i, when it is revolute the variable is θ_i. Using these parameters, the transformation matrix from frame i to frame *i-1* is the following:

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

The variable q_i is the joint variable which depends on the type of joints, as said previously. Once all the transformation matrices have been defined, they are combined to obtain the direct kinematic function:

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

3.2.2 CNC machine Direct Kinematics

The Denavit-Hartenberg convention was applied to the part of the CNC machine that moves the cutting tool. In figure 3.5, it is shown the CNC machine kinematic chain with the reference frames obtained applying the theory previously exposed. This part is composed by two prismatic joints, thus the choice of the z axes is not univocal. They were chosen along their respective direction of motion.

From these reference frames, the Denavit-Hartenberg parameters was extrapolated. Having two prismatic joints, the joint variables are d_1 and d_2. The obtained parameters are shown in Table 3.1:

DH parameters	a_i	α_i	d_i	θ_i
Joint 1	0	90°	d_1	90°
Joint 2	0	−90°	d_2	90°

Table 3.1: Denavit-Hartenberg parameters of CNC machine first part

Substituting these parameters in the formula previously exposed, the following matrices were obtained:

$$A_1^0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

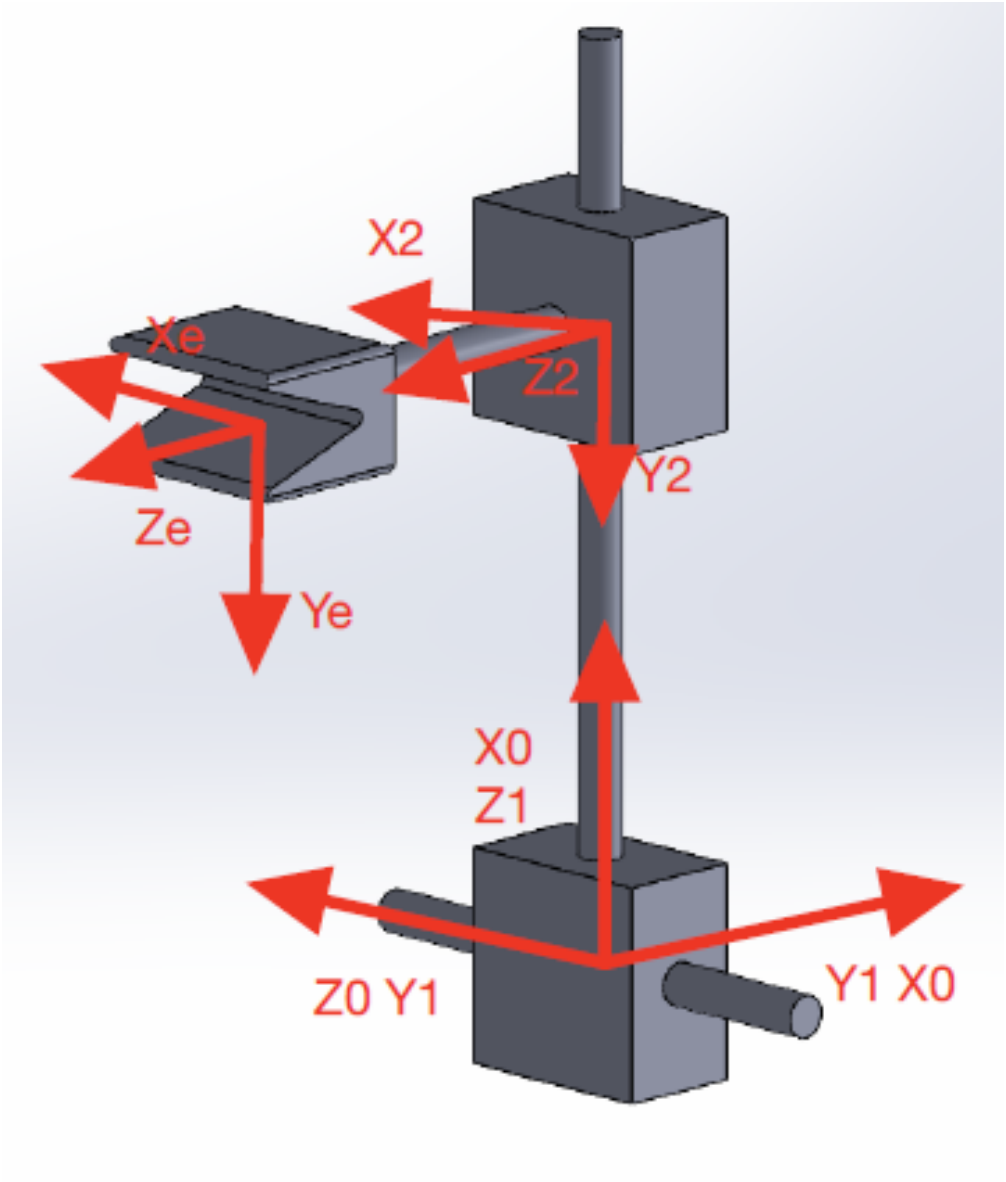

Figure 3.5: Model of the CNC machine part 1

$$A_2^1 = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^0 = A_1^0 A_2^1 = \begin{bmatrix} 0 & -1 & 0 & d_2 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For our purpose, it is necessary to have a reference frame on the end-effector tool base. Consequently, the reference frame $x_e - y_e - z_e$ was introduced at a distance $\overline{d}$ along the axes z_2. The final matrix obtained is:

$$T_e^0 = \begin{bmatrix} 0 & -1 & 0 & d_2 \\ 0 & 0 & -1 & -\overline{d} \\ 1 & 0 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This last matrix allows to compute the position of the end-effector knowing the values of the joint variables. However, to replicate the motion of the CNC machine, the method employed needs a matrix called Jacobian. This matrix is found using the differential kinematic analysis which is explained in the next section.

3.3 Differential Kinematics

The second step of our analysis of the CNC machine is the differential kinematics analysis. It provides a relation between joint velocities and the corresponding end-effector linear and angular velocities. This relation is obtained through a matrix, termed Jacobian, which depends on the manipulator configuration. There are two different types of Jacobians: Geometric Jacobian and Analytical Jacobian. The first is obtained from the end-effector pose expressed as homogeneous transformation matrix; the second from the minimal representation of the end-effector pose. For our purpose, the Geometric Jacobian was chosen.
Defining $\dot{p}_e$ the end-effector linear velocity, ω_e its angular velocity and $\dot{q}$ the joint velocities, the following relation can be written as [14]:

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

$$\omega_e = J_O(q)\dot{q}$$

Where J_P is the is part of the Geometric Jacobian which contains the contribution of the joint velocities to the end-effector linear velocity and J_O to the angular velocity. Assembling these matrices, the Geometric Jacobian (6xn) is obtained. Where n is the number of joints. In compact form, the previous relations become:

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

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

Avoiding the details about the theory to compute the Geometric Jacobian, the final results are exposed. The linear velocity $\dot{p}_e$ can be expressed as:

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

Where $j_{Pi}\dot{q}_i$ is the contribution of the velocity of joint i to the end-effector linear velocity when the other joints do not move. This contribution depends on the type of joint.
In the case of a prismatic joint which have joint variable $q_i = d_i$:

$$\dot{q}_i j_{Pi} = \dot{d}_i z_{i-1}$$

and then

$$j_{Pi} = z_{i-1}$$

In the case of a revolute joint which have joint variable $q_i = \theta_i$:

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

and then

$$j_{Pi} = z_{i-1} \times (p_e - p_{i-1})$$

Similarly, the angular velocity of the end-effector ω_e can be expressed as:

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

It is possible to compute the contribution to the angular velocity for the two types of joints.
In the case of the prismatic joint:

$$\dot{q}_i j_{Oi} = 0$$

and then

$$j_{Oi} = 0$$

In the case of a revolute joint:

$$\dot{q}_i j_{Oi} = \dot{\theta}_i z_{i-1}$$

and then

$$j_{Oi} = z_{i-1}$$

Assembling all the found contributions, the Geometric Jacobian is:

$$J = \begin{bmatrix} j_{P1} & & j_{Pn} \\ & \cdots & \\ j_{O1} & & j_{On} \end{bmatrix}$$

where:

$$\begin{bmatrix} j_{Pi} \\ j_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} & \text{for a prismatic joint} \\ \begin{bmatrix} z_{i-1} \times (p_e - p_{i-1}) \\ z_{i-1} \end{bmatrix} & \text{for a revolute joint} \end{cases}$$

The parameters z_{i-1}, p_e and p_{i-1} depends on the joint variables. In particular:

- z_{i-1} is the third column of the rotation matrix R^0_{i-1}:

  $$z_{i-1} = R^0_1(q_1) \dots R^{i-2}_{i-1}(q_{i-1}) z_0$$

  where z_0=[0 0 1]^T.

- p_e is composed by the first three elements of the fourth column of the homogeneous transformation matrix T^e_0. These elements can be extracted from:

  $$\tilde{p}_e = A^0_1(q_1) \dots A^{n-1}_n(q_n) \tilde{p}_0$$

  where $\tilde{p}_0$=[0 0 0 1]^T.

- p_{i-1} is composed by the first three elements of the fourth column of the homogeneous transformation matrix T_0^{i-1}. These elements can be extracted from:

$$\tilde{p}_{i-1} = A_1^0(q_1)\dots A_{i-1}^{i-2}(q_{i-1})\tilde{p}_0$$

Applying this method to the CNC machine structure considered, the following matrices were obtained:

$$z_0{=}[0\ 0\ 1]^T$$

$$z_1{=}[1\ 0\ 0]^T$$

$$J = \begin{bmatrix} z_0 & z_1 \\ 0 & 0 \end{bmatrix}$$

$$J = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

From This Jacobian it is possible to extract a full rank matrix, obtaining the following final equation:

$$V_e = \begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = J(q)\dot{q} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

3.4 Inverse Kinematic Algorithm

The direct kinematics and differential kinematics analyses previously exposed have been employed to find the solution of the inverse kinematics problem. It consists in computing the values of the joint variables ($\mathbf{q}$(t)) which allow to achieve a desired configuration of the end-effector. There are many solutions for the inverse kinematics problem, among them the inverse differential kinematics method was chosen. It allows, given the desired trajectory for the end-effector, expressed as v_e and the initial conditions on position and orientation, to compute the corresponding joint trajectory ($\mathbf{q}$(t),$\dot{\mathbf{q}}$(t)). Starting from the differential kinematics equation [14]:

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

The joint velocities $\dot{\mathbf{q}}(\mathrm{t})$ can be simply obtained through the inversion of the Jacobian:

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

To apply this formula, it is necessary that the Jacobian is square and full rank to be invertible. If this does not occur, a full rank matrix is extracted from the Jacobian and, if this matrix is not square, it is used the right pseudo-inverse of this matrix. However, this is not our case, because the extracted matrix from the Jacobian is square, thus it is invertible. After the computation of $\dot{\mathbf{q}}(\mathrm{t})$, knowing the manipulator initial posture $\mathbf{q}(0)$, the joint position is computed as:

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

In discrete time, this integral can be solved through numerical techniques, the simplest is the Euler integration method. Given an integration interval δt, knowing the joint positions and velocities at time t_k, the joint positions at time $t_{k+1} = t_k + \delta\mathrm{t}$ is:

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

Substituting $\dot{q}(t_k)$ in discrete form, the following relation is obtained:

$$q(t_{k+1}) = q(t_k) + J^{-1}(q(t_k))v_e(t_k)\delta t$$

Unluckily, the numerical integration introduces some inaccuracies in the computed value of the joint position. Thus, the end-effector pose obtained does not perfectly coincide with the desired one. This problem can be solved using an algorithm that considers the error and reduces it. As first step, the operational space error is defined as:

$$e = x_d - x_e$$

Where x_d is the desired pose of the end-effector and x_e is the effective one. Considering the time derivative of the previous equation:

$$\dot{e} = \dot{x}_d - \dot{x}_e$$

Considering the differential kinematics equation, it becomes:

$$\dot{e} = \dot{x}_d - J(q)\dot{q}$$

This kind of algorithm needs the use of the Analytical Jacobian because the operational space quantities are involved in the process. However, in our case, the use of the Geometric Jacobian previously computed is correct because the Geometric and the Analytical Jacobians differ only for the part that concerns the

rotation, instead, the part about the linear velocity is the same. The considered part of the CNC machine has only two prismatic joints, thus the movements and the velocities are only linear, for this reason the computed Geometric Jacobian concerns only the linear velocity.
Now, considering that the Jacobian is invertible and choosing:

$$\dot{q} = J^{-1}(q)(\dot{x}_d + Ke)$$

It is obtained the equivalent linear system:

$$\dot{e} + Ke = 0$$

With K is a positive definite diagonal matrix to achieve an asymptotically stable system. The eigenvalues of matrix K influence the rate at which the error converges to zero, by increasing the eigenvalues, the rate increases. However, the eigenvalues have a limit for the maximum values that they can assume because the algorithm is implemented in discrete time. This limit is influenced by the sampling time and the asymptotic stability of the system is guaranteed only under this value. The final scheme of this algorithm is shown in figure 3.6.

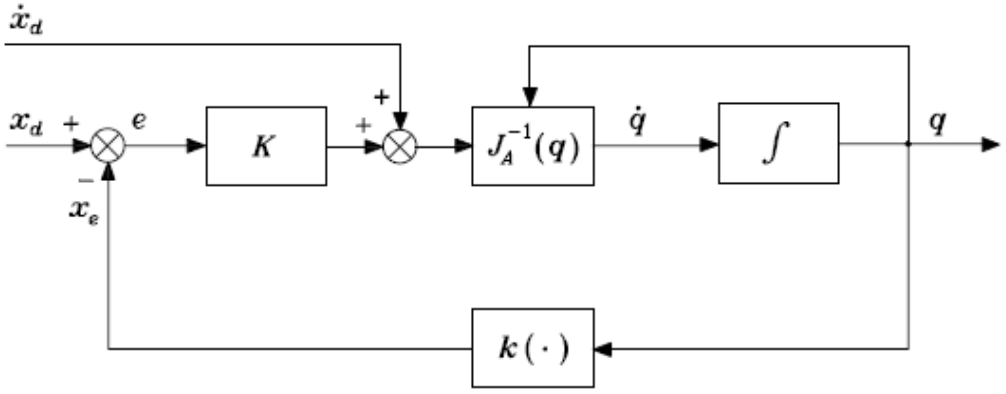

Figure 3.6: Block scheme of the inverse kinematics algorithm

This algorithm was employed on MATLAB to develop a simulation of the CNC machine part which moves the cutting tool. The model produced (figure 3.7) is a precise reproduction of the previous block scheme. The Inverse_Jacobian block should invert the Jacobian at each time instant because it should vary in relation to the manipulator pose. However, in our case the Jacobian is always constant. The Direct_Kinematics block uses the homogeneous transformation matrix to compute the effective position of the end-effector x_e employing the joint variables at each

time instant. The inputs of the system are the desired position x_d and the desired velocity $\dot{x}_d$ of the end-effector. Moreover, the initial position of the joint variables is provided to the integration block. The output of the system is the effective position of the end-effector, indeed we are interested in the movements of the cutting tool.

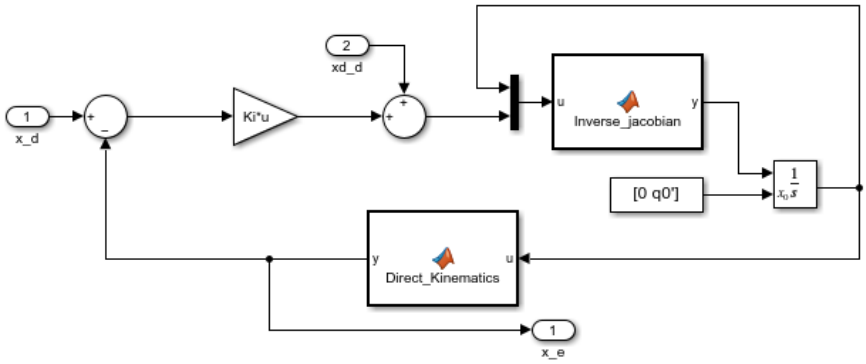

Figure 3.7: Block scheme of the inverse kinematics algorithm on MATLAB

3.5 Workpiece position and orientation

The CNC machine, as said previously, was considered divided in two parts that acts independently. The previous sections described the analysis of the first part which moves the cutting tool. In this section instead, the second part of the CNC machine is analysed. It was studied through the direct kinematic theory, in order to find the homogeneous transformation matrix associated to the workpiece.
The second part of the CNC machine moves and orientate the workpiece. Its kinematic chain scheme with the associated reference frames obtained from the Denavit-Hartenberg convention is shown in figure 3.8. It is composed by a prismatic joint that moves the workpiece along the axis z_0, bringing it closer to the cutting tool, and two revolute joints that orientate the piece. The revolute joints were considered coincident and the reference frames were placed consequently, in the previous figure they are not represented coincident to have a clear scheme of the kinematic chain. From these reference frames, the Denavit-Hartenberg parameters was extrapolated. Having one prismatic joint and two revolute ones, the joint variables are d_1, θ_2 and θ_3. The resulting parameters are shown in Table 3.2.

Substituting these parameters in the formula previously exposed, the following matrices were obtained:

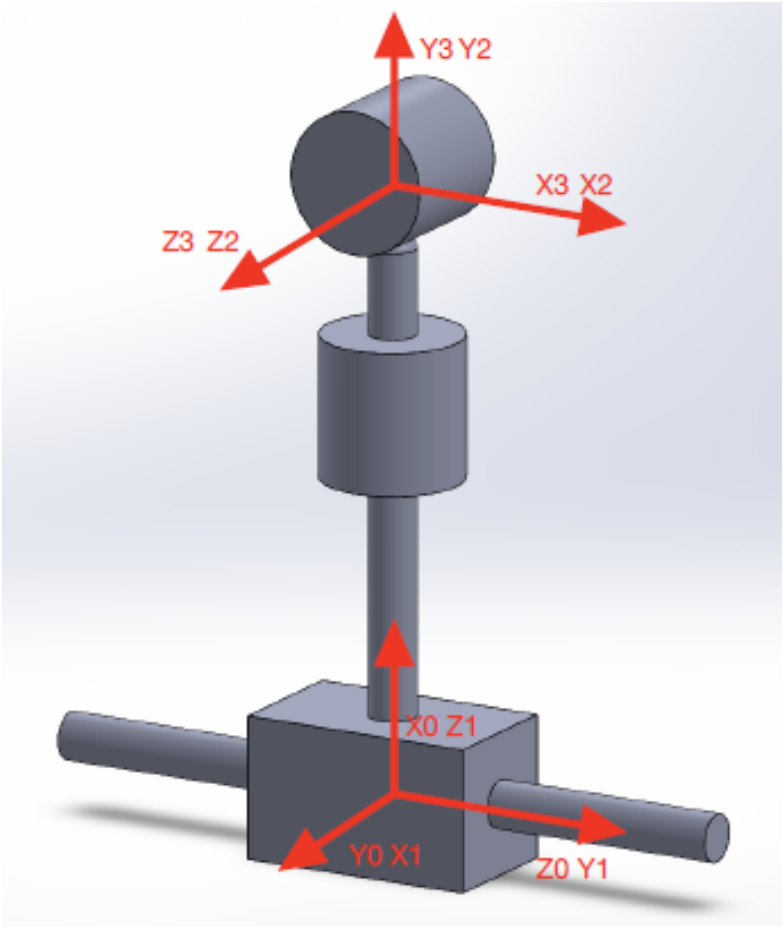

Figure 3.8: Model of the CNC machine part 2

DH parameters	a_i	α_i	d_i	θ_i
Joint 1	0	90°	d_1	90°
Joint 2	0	90°	$\overline{a}$	θ_2
Joint 3	0	0	0	θ_3

Table 3.2: Denavit-Hartenberg parameters of CNC machine second part

$$A_1^0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2^1 = \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & \overline{a} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2 = \begin{bmatrix} c_3 & -s_3 & 0 & 0 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^0 = A_1^0 A_2^1 A_3^2 = \begin{bmatrix} s_3 & c_3 & 0 & \overline{a} \\ c_2 c_3 & -c_2 s_3 & s_2 & 0 \\ s_2 c_3 & -s_2 c_3 & -c_2 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For our purpose, it is necessary to have the coordinate of the workpiece and those of the cutting tool expressed with respect to the same reference frame. Consequently, a base reference frame was considered, expressing the coordinates of the workpiece with respect to it. To simplify our case, as base frame was considered the frame $x_0 - y_0 - z_0$ of the first part of the CNC machine that is the frame used to express the coordinates of the cutting tool. Consequently, considering that the base frame and the actual frame $x_0 - y_0 - z_0$ are both fixed reference frame, the transformation matrix between them was computed:

$$T_0^b = \begin{bmatrix} 1 & 0 & 0 & -\overline{a}_1 \\ 0 & 0 & 1 & -\overline{a}_2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

To obtain the coordinates of the workpiece expressed with respect to the base frame, this matrix was combined with matrix T_3^0:

$$T_3^b = T_0^b T_3^0 = \begin{bmatrix} s_3 & c_3 & 0 & \overline{a} - \overline{a}_1 \\ s_2 c_3 & -s_2 s_3 & -c_2 & d_1 - \overline{a}_2 \\ -c_2 c_3 & c_2 s_3 & -s_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The purpose of this thesis work is to identify the points of the end-effector tool that interact with the workpiece. This occurs when these points are under the work surface of the piece. Therefore, only the work surface was considered, employing

the previous matrix to obtain its position and orientation after the variations of the joint variables of the second part of the CNC machine.
The code was created on MATLAB, setting the parameters shown in table 3.3.

$\overline{a}$	6
$\overline{a}_1$	2
$\overline{a}_2$	8

Table 3.3: Constant parameters of the matrix T_3^b

To compute the coordinates of the rotating points that compose the work surface, the starting point was to find its vertices position. The workpiece is positioned on a mobile platform. The matrix T_3^b express the position and orientation of the reference frame $x_3 - y_3 - z_3$, whose origin coincide with the centre of the platform, with respect to the base frame. However, the work surface is not on the centre of the platform, it is translated along axis x_3. Moreover, the coordinates along axis y_3 starts from zero because, obviously, the workpiece can not be under the platform. Taking into account that and considering a cubic workpiece with sides long 10, the vertices coordinates with respect to $x_3 - y_3 - z_3$ with null joint variables are shown in table 3.4.

x_3	y_3	z_3
5	0	5
5	0	-5
5	10	5
5	10	-5

Table 3.4: Vertices coordinates with respect to reference frame $x_3 - y_3 - z_3$

Using these coordinates and the matrix T_0^b, the coordinates of the moved vertices of the work surface with respect to base frame were computed. Using this method, all the coordinates of the points that compose the work plane were computed. Subsequently, some figures that show the results are exposed. In the first three, the joint variables values vary individually; in the last one, they vary simultaneously. In figure 3.9, the original work surface (red) and the rotated one (blue) are shown. The joint variables were set with the following values: $d_1 = 0$, $\theta_2 = \frac{\pi}{2} + \frac{\pi}{10}$, $\theta_3 = 0$.

In the values of θ_3, the value $\frac{\pi}{2}$ accounts for a mismatch between the reference frames, thus only the term $\frac{\pi}{10}$ introduce a rotation around the axis y_3 of the last frame of the CNC machine second part.

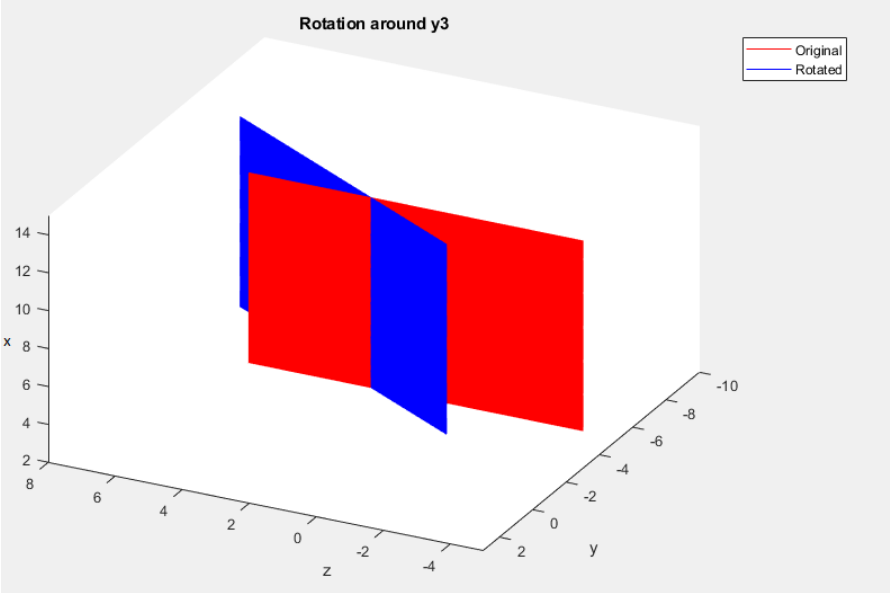

Figure 3.9: Work surface in the original pose (red) and after a rotation around axis y_3 (blue)

In figure 3.10, the joint variables were set with the following values: $d_1 = 0$, $\theta_2 = \frac{\pi}{2}$, $\theta_3 = -\frac{\pi}{10}$. Consequently, it shows only a rotation around axis z_3. Instead, in figure 3.11, the joint variables have these values: $d_1 = 5$, $\theta_2 = \frac{\pi}{2}$, $\theta_3 = 0$. It displays only a translation along axis x_3.

In the previous figures, the joint variables were considered individually and the set values were larger than usual in order to highlight the movements. However, the rotation angles always assume smaller values. In figure 3.12, the joint variables were considered together and their values were set in a plausible way: $d_1 = 2$, $\theta_2 = \frac{\pi}{2} + \frac{\pi}{45}$, $\theta_3 = -\frac{\pi}{45}$. The surface is translated along axis x_3 and rotated by 4 degrees around both the z_3-axis and the y_3-axis.

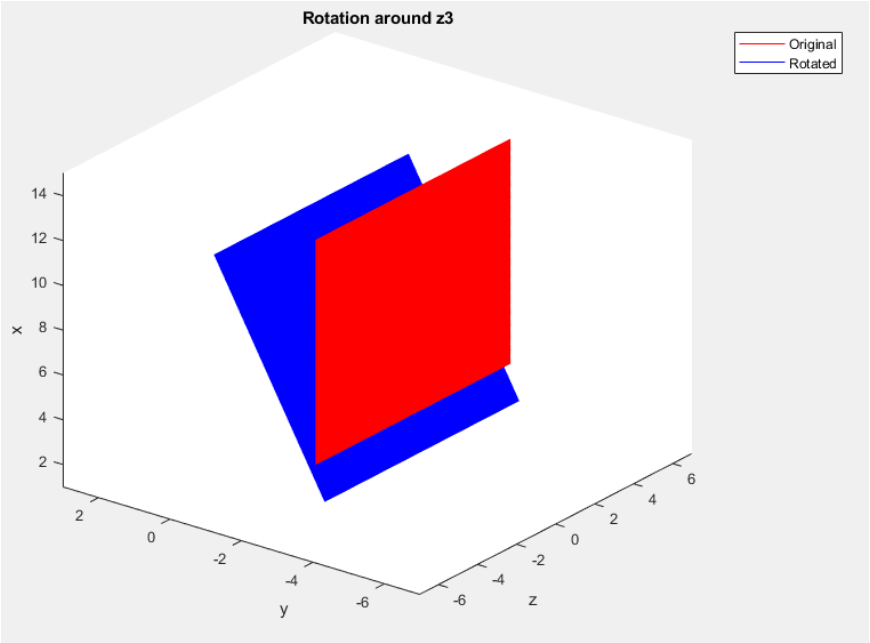

Figure 3.10: Work surface in the original pose (red) and after a rotation around axis z_3 (blue)

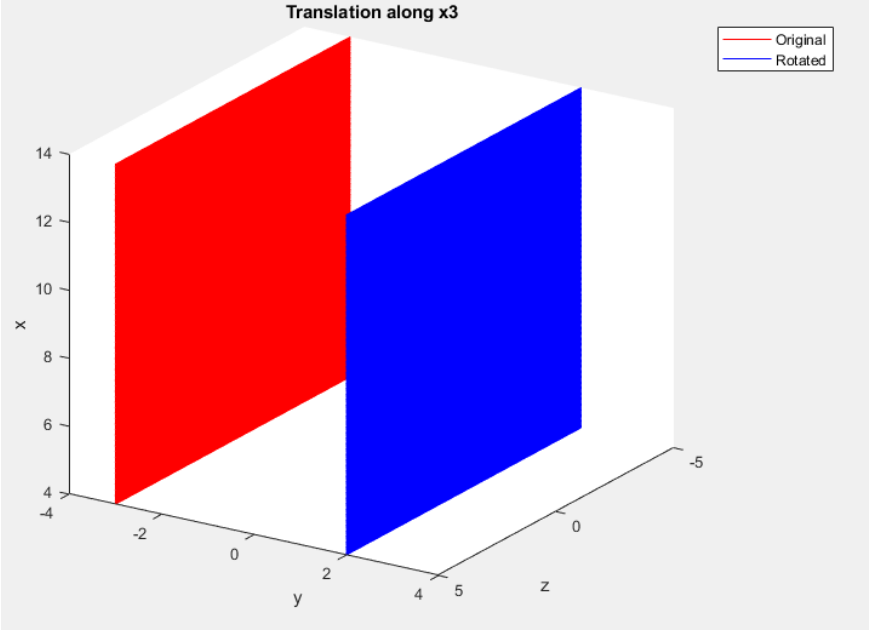

Figure 3.11: Work surface in the original pose (red) and after a translation along axis x_3 (blue)

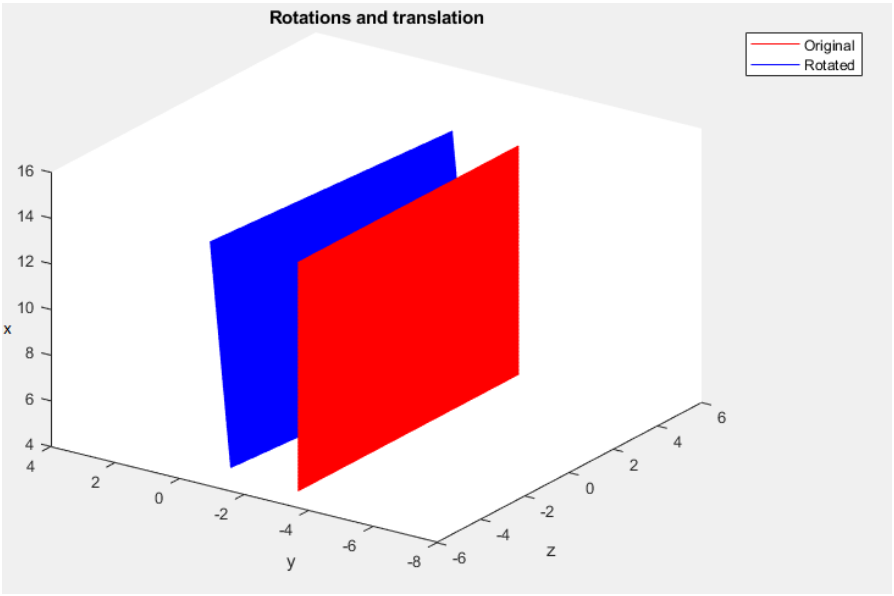

Figure 3.12: Work surface in the original pose (red) and after translation and rotations (blue)

Chapter 4

Trajectory Planning

The developed model of the first part of the CNC machine needs as inputs the desired position and velocity of the end-effector. In this chapter it is explained how these inputs are generated suitably. The trajectory planning is the part of Robotics that study the method to generate a time sequence of values as reference input for the system in order to achieve the desired position. After an overview about the concept of trajectory, the problem of generating a trajectory in the operational space is exposed and, subsequently, it is shown how a rectilinear trajectory and a circular one were computed. Lastly, the results obtained applying these inputs to the CNC machine model are illustrated.

4.1 Trajectory

A manipulator must be able to move from an initial pose to a desired one. However, the transition must be gradual to respect the physical limits of the actuators and to respect some possible working condition. Therefore, it is necessary to generate a suitable smooth trajectory. To define the concept of trajectory it is necessary to explain what a path is. It is the locus of points that the manipulator has to follow to obtain the desired pose. It describes the motion from a geometrical viewpoint. Instead, a trajectory is a path combined with a timing law, thus to each point of the path is associated a velocity and, sometimes, an acceleration. The trajectory is obtained through interpolation functions. The user choices some important parameters as the extremal points, the type of function used, the maximum velocity and acceleration, etc. Using these data, the trajectory planning algorithm produce a trajectory that respect these constraints. There are two types of trajectories:

- **Joint space trajectory**: this kind of trajectory is composed by the values that joint variables and joint velocities have to assume during the motion. The initial and final pose of the end-effector are usually assigned in the operational

space; thus, it is necessary an inverse kinematic algorithm to convert these parameters into joint variables values. A trajectory generated in the joint space makes easier to deal with singular configuration and it usually involves a low computationally effort.

- **Operational space trajectory**: it is composed by the pose and the velocity values that the end-effector has to assume during the motion. It is used when the end-effector has to follow a specified path because of some constraints along it, as the avoidance of an obstacle. However, to plan a trajectory in the operational space is more difficult.

For our purpose, the operational space trajectory was chosen because a CNC machine must follow a precise path during the cutting operation and, from this viewpoint, this technique ensures better results. Moreover, it allows to establish the velocity of the end-effector during the motion. In our case, the CNC machine is very simple, the first part, for which the trajectory must be planned, is composed only by two prismatic joint. Thus, also a joint space trajectory would have provided good results. However, we decided to not employ it and to develop the simulator in a generic manner; so, in the future, it would be possible to use it with others CNC machine models that could be more complex.

4.1.1 Operational space trajectory

To explain how to plan a trajectory in operational space, it is necessary to introduce the concept of parametric description of paths in the space. Considering a (3x1) vector $\mathbf{p}$ and a continuous vector function $\mathbf{f}(\sigma)$ defined in the interval $[\sigma_i, \sigma_f]$, the following equation is defined [14]:

$$\text{p= f}(\sigma)$$

Where the sequence of values assumed by $\mathbf{p}$ as σ varies is termed path Γ in space. The previous relation defines its parametric representation with respect to the parameter σ. Starting from σ_i and increasing the value of σ, the point $\mathbf{p}$ moves along the path in a specific direction which is defined by the function $\mathbf{f}(\sigma)$. Considering the point p_i of a generic open path Γ (figure 4.1) and its generic point $\mathbf{p}$, they identify an arc of Γ for which they are the extremes, the length of this arc is called arc length s and its value is positive when p follows p_i along the direction of the path, it is negative in the other case. Using s as parameter, it is possible to obtain a different parametric representation of the path Γ:

$$\text{p= f(s)}$$

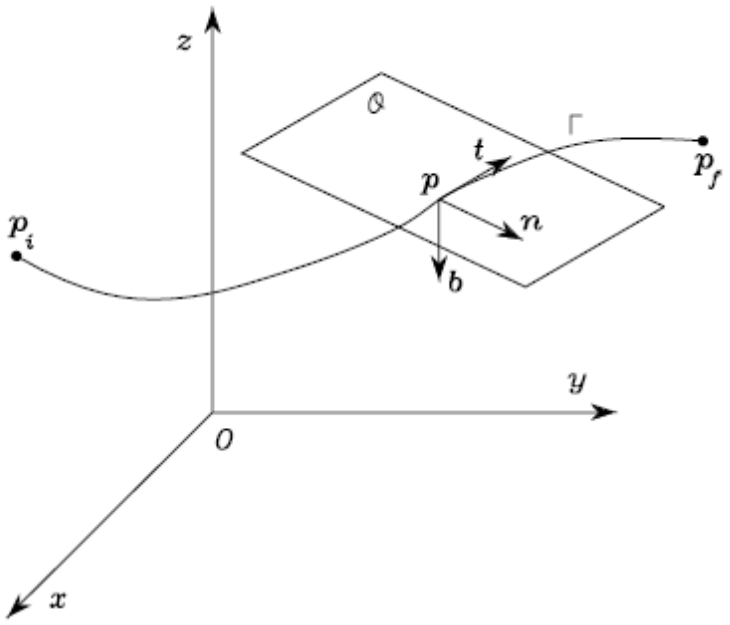

Figure 4.1: Parametric representation of a path in the space

Using this parametric representation, three unitary vectors are identified by each generic point p of the path Γ:

- Tangent unit vector t which is tangent to the path Γ in point p and its direction is oriented along the path direction;
- Normal unit vector n which is normal to t and it is located in the osculating plane O (figure 4.1);
- Binormal unit vector b which is defined according to the right-hand rule to complete the frame (t, n, b).

These unit vectors depend on the arc length s according to the following relations:

$$t = \frac{dp}{ds}$$

$$n = \frac{1}{\left\| \frac{d^2p}{ds^2} \right\|} \frac{d^2p}{ds^2}$$

$$b = t \times n$$

The goal of a trajectory in the operational space, as said previously, is to generate a function p(t) which allows the end-effector to reach the desired pose in a time t_f following a specific path and respecting a specific motion law. Considering a relation p=f(s) which describes the path Γ that has as extremes the initial point p_i and the final point p_f, thus the arc length value at t=0 is s=0 and at t=t_f is s=s_f (path length). The arc length varies from these two values in relation to time, thus, the function s(t) describes the timing law along the path. It is chosen by the developer among different functions, on the base of some specific requirement or constraint. In our case, a timing law with trapezoidal velocity profile was chosen.

4.1.2 Timing law with trapezoidal velocity profile

This timing law was chosen because it permits to set a cruise velocity for the movement of the end-effector. In metal cutting operations, the cutting tool moves with respect to the workpiece with a constant cutting speed, thus this time law approximates suitably the tool behaviour. It is assumed that the initial velocity is null, successively a constant acceleration acts for a time t_c, thus the velocity increases with linear behaviour and it reaches the set cruise velocity at time t=t_c; after this time instant, the velocity maintains the set value until a constant deceleration acts at time t=$t_f - t_c$ and it cause a linear decreasing of the velocity until it reaches the null value at time t=t_f (figure 4.2). The behaviour descripted implies that the accelerations have the same time duration, producing a function with symmetrical profile [14].

The timing law with trapezoidal velocity profile is composed by the following sequence of polynomials:

$$s(t) = \begin{cases} \frac{1}{2}\ddot{p}_c t^2 & 0 \leq t \leq t_c \\ \ddot{p}_c t_c (t - \frac{t_c}{2}) & t_c < t \leq t_f - t_c \\ s_f - \frac{1}{2}\ddot{p}_c (t_f - t)^2 & t_f - tc < t \leq t_f \end{cases}$$

Where $\ddot{p}_c$ is the constant acceleration, its modulus is the same for the positive acceleration and the negative one, and s_f is the value that the arc length must have at t=t_f.

To compute the parameters $\ddot{p}_c$ and t_c is necessary to define the previous function. Firstly, the initial and final position of the end-effector, the time t_f and the desired cruise velocity are chosen. Afterwards, it is necessary to check if the chosen values satisfy the following constraint:

$$\frac{\|s_f\|}{t_f} < |\dot{p}_c| \leq 2\frac{\|s_f\|}{t_f}$$

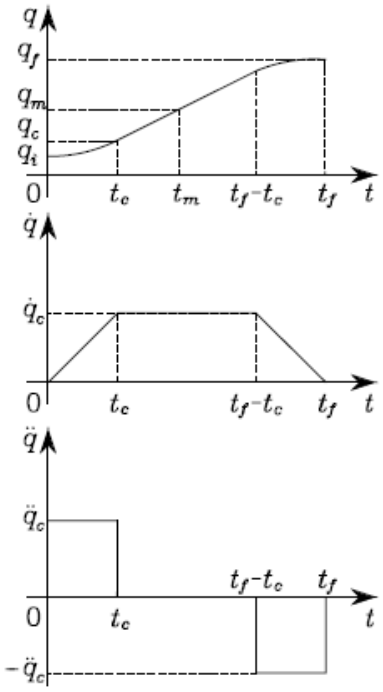

Figure 4.2: Position, velocity and acceleration of the timing law with trapezoidal velocity profile

If it is not satisfied, this means that the values do not consent to obtain a function with the desired behaviour and, thus, it is necessary to change them. Then, t_c and $\ddot{p}_c$ are computed using the following relation:

$$t_c = \frac{\dot{p}_c t_f - s_f}{\dot{p}_c}$$

$$\ddot{p}_c = \frac{\dot{p}_c^2}{\dot{p}_c t_f - s_f}$$

These values allow the generation of the sequence of polynomials previously exposed. The theoretical notions defined above were used to plan a trajectory with a rectilinear path and another with circular path.

4.2 Rectilinear trajectory

The first trajectory planned connects the initial point p_i to the point p_f with a rectilinear path whose parametric representation is:

$$p(s) = p_i + \frac{s}{||p_f - p_i||}(p_f - p_i)$$

When the arc length s is equal to zero, p(0)=p_i; instead when s=$||p_f - p_i||$, p($||p_f - p_i||$)=p_f. The velocity is equal to:

$$\dot{p} = \frac{\dot{s}(p_f - p_i)}{||p_f - p_i||} = \dot{s}t$$

Where t is the tangent unit vector.
Considering the timing law with trapezoidal velocity profile, the value that the arc length s assumes at time t=t_f is $s_f = ||p_f - p_i||$. Thus, the previous relations and the constraint become:

$$\frac{\|p_f - p_i\|}{t_f} < |\dot{p}_c| \leq 2\frac{\|p_f - p_i\|}{t_f}$$

$$t_c = \frac{\dot{p}_c t_f - ||p_f - p_i||}{\dot{p}_c}$$

$$\ddot{p}_c = \frac{\dot{p}_c^2}{\dot{p}_c t_f - ||p_f - p_i||}$$

$$s(t) = \begin{cases} \frac{1}{2}\ddot{p}_c t^2 & 0 \leq t \leq t_c \\ \ddot{p}_c t_c (t - \frac{t_c}{2}) & t_c < t \leq t_f - t_c \\ \|p_f - p_i\| - \frac{1}{2}\ddot{p}_c (t_f - t)^2 & t_f - tc < t \leq t_f \end{cases}$$

These relations were implemented on MATLAB in discrete time domain. To show the obtained results, two examples are proposed: a trajectory planning in 2-D and another in 3-D.

4.2.1 Rectilinear trajectory planning in 2-D

Considering that the first part of CNC machine is composed by two prismatic joint, it can move its end-effector only in two directions (along x-axis and z-axis); thus, the trajectories planned for this model has 2-D paths. For this reason, an example about the 2-D trajectory planning is exposed in this section. The parameters contained in table 4.1 were used to project the trajectory.

Using these parameters, the time t_c and the acceleration $\ddot{p}_c$ were computed:

$$t_c = 0.4277 \text{ s}$$
$$\ddot{p}_c = 14.0295 \ cm/s^2$$

p_i	(5, 7)
p_f	(10, 15)
t_i	0 s
t_f	2 s
$\dot{p}_c$	6 cm/s

Table 4.1: Parameters employed in a 2-D rectilinear trajectory planning

The arc length found is:

$$s(t) = \begin{cases} 7.0148t^2 & 0 \leq t \leq 0.4277 \\ 6(t - 0.2138) & 0.4277 < t \leq 1.5723 \\ 9.434 - 7.0148(2 - t)^2 & 1.5723 < t \leq 2 \end{cases}$$

which was used to compute p(t) and $\dot{p}(t)$. Then, a MATLAB function was created, obtaining the results shown in figures 4.3, 4.4 and 4.5:

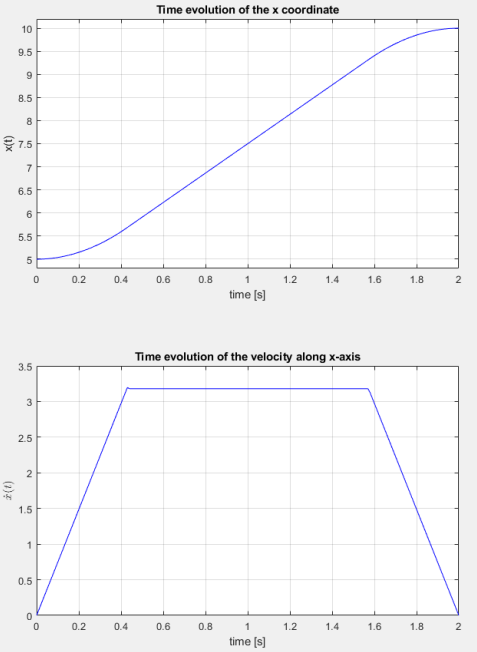

Figure 4.3: Time evolution of the x coordinate and velocity along x-axis in a 2-D rectilinear trajectory planning

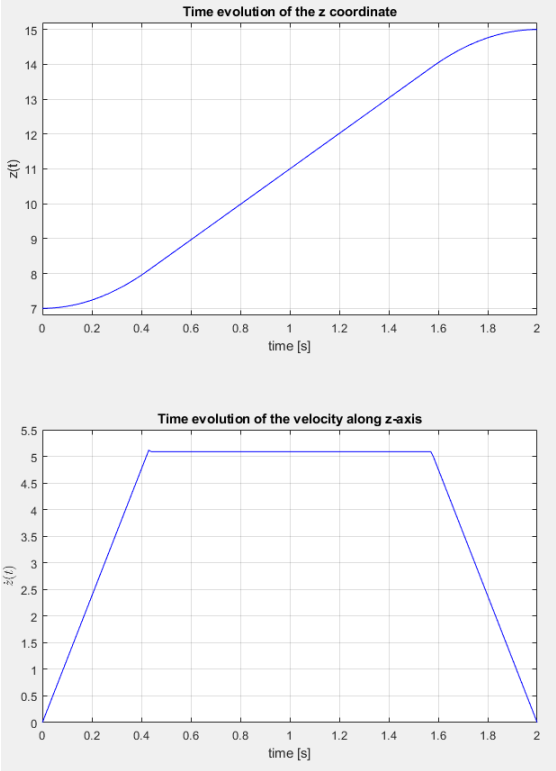

Figure 4.4: Time evolution of the z coordinate and velocity along z-axis in a 2-D rectilinear trajectory planning

In figure 4.3 and 4.4 is shown the time evolution of x and z coordinates and the evolution of velocities along these axes. It is possible to observe that each coordinate reaches the final value set and the functions have the desired behaviour. In figure 4.5, it is shown the evolution of x coordinate with respect to z coordinate and the evolution of the velocity along the direction of motion. From the first graph, it is possible to notice that the path between the initial and the final point is rectilinear as desired. In the second graph, the velocity has the correct behaviour and it reaches the cruise velocity set $\dot{p}_c$ along the direction of motion.

These results shows that the developed algorithm to plan a rectilinear trajectory works properly in the 2-D case.

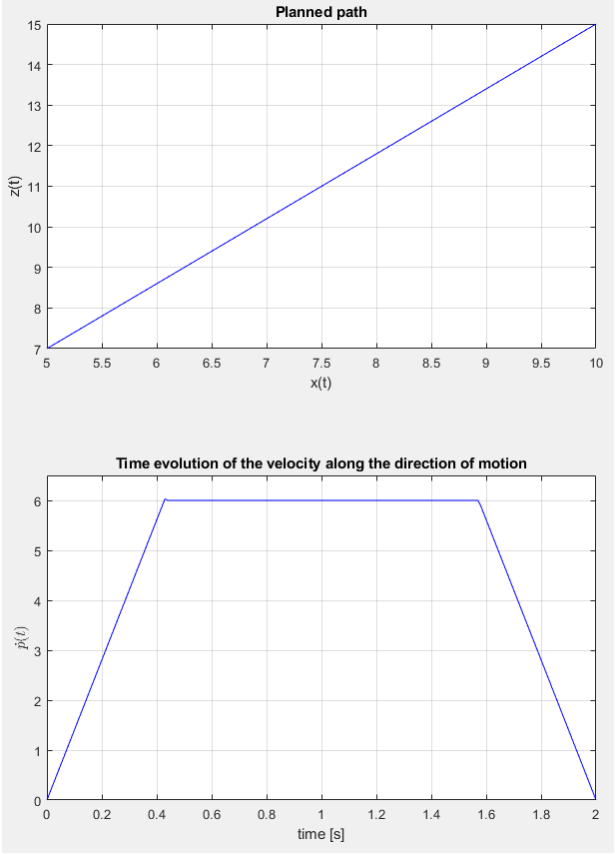

Figure 4.5: Planned path and time evolution of the velocity along the direction of motion in a 2-D rectilinear trajectory planning

4.2.2 Rectilinear trajectory planning in 3-D

In the future, the simulator on which I worked could be used with other CNC machine models, thus, the trajectory planning algorithm was tested also in the 3-D case. In this section, an example is exposed. The parameters employing are contained in table 4.2.

Using these parameters, the time t_c and the acceleration $\ddot{p}_c$ were computed:

$$t_c = 0.252 \text{ s}$$

$$\ddot{p}_c = 23.8109 \ cm/s^2$$

The arc length found is:

p_i	(6, 8, 2)
p_f	(11, 14, 9)
t_i	0 s
t_f	2 s
$\dot{p}_c$	6 cm/s

Table 4.2: Parameters employed in a 3-D rectilinear trajectory planning

$$s(t) = \begin{cases} 11.9055t^2 & 0 \leq t \leq 0.252 \\ 6(t - 0.126) & 0.252 < t \leq 1.748 \\ 10.4881 - 11.9055(2 - t)^2 & 1.748 < t \leq 2 \end{cases}$$

It was used to compute p(t) and $\dot{p}(t)$. In figure 4.6, 4.7 and 4.8 is shown the time evolution of x, y and z coordinates and the evolution of velocities along these axes. It is possible to observe that each coordinate reaches the final value set and the functions have the desired behaviour.

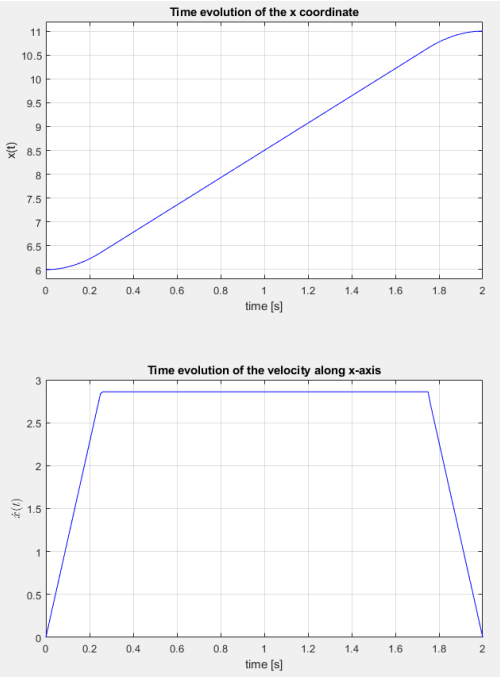

Figure 4.6: Time evolution of the x coordinate and velocity along x-axis in a 3-D rectilinear trajectory planning

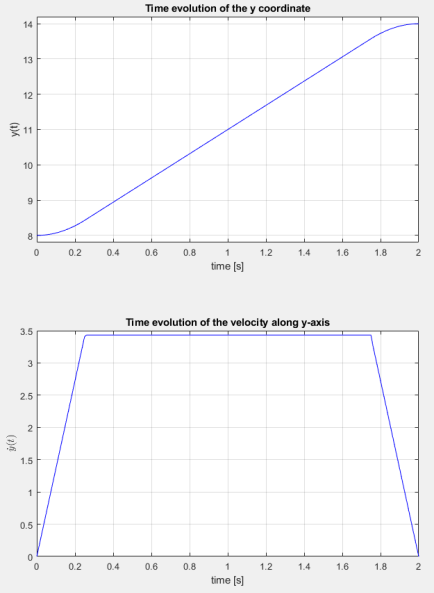

Figure 4.7: Time evolution of the y coordinate and velocity along y-axis in a 3-D rectilinear trajectory planning

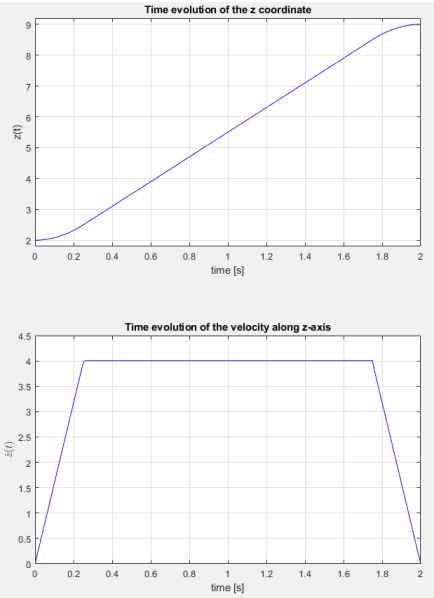

Figure 4.8: Time evolution of the z coordinate and velocity along z-axis in a 3-D rectilinear trajectory planning

In figure 4.9, it is shown the 3-D path and the evolution of the velocity along the direction of motion. From the first graph, it is possible to notice that the path between the initial and the final point is rectilinear as desired. In the second graph, the velocity has the correct behaviour and it reaches the cruise velocity set $\dot{p}_c$ along the direction of motion.

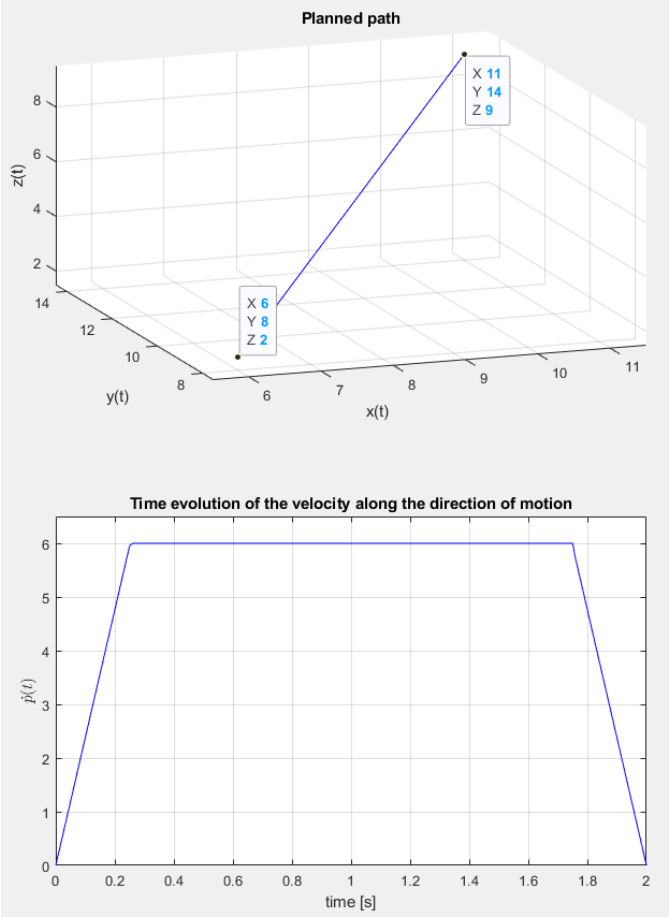

Figure 4.9: Planned path and time evolution time of the velocity along the direction of motion in a 3-D rectilinear trajectory planning

These results shows that the algorithm developed to plan a rectilinear trajectory works properly also in the 3-D case. However, it involves only the position of the end-effector, the orientation was not considered because of the structure of the CNC machine taken in exam. Consequently, if in the future the simulator is used with another CNC machine model, according to its structure, it is possible that the part about the orientation of the trajectory planning will be integrated in this algorithm.

4.3 Circular trajectory

To plan a circular trajectory, it is essential to define some parameters. Considering the closed path Γ which is a circumference (figure 4.10), the following parameters are used to define it [14]:

- the unit vector r along the axis of the circumference;
- the position vector d of a point along the axis of the circumference;
- the position vector p_i of a point on the circumference.

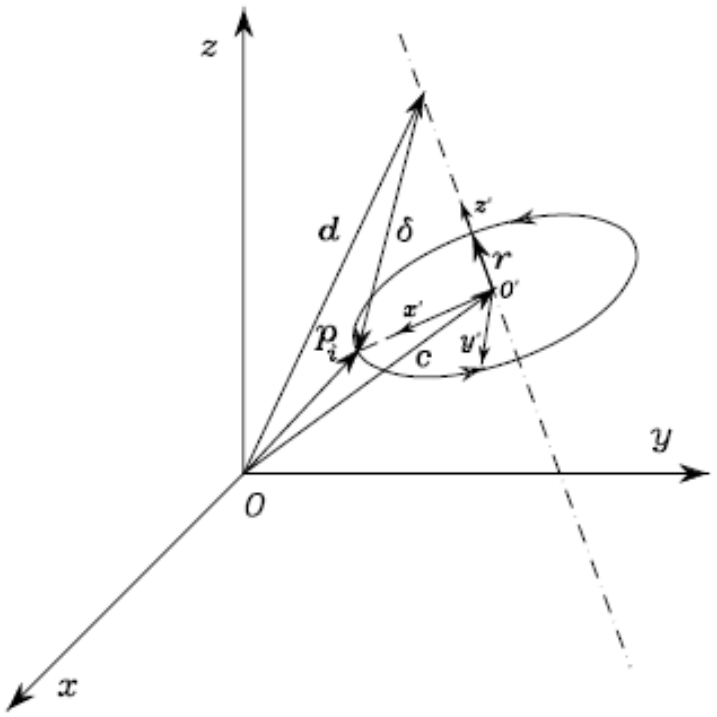

Figure 4.10: Parametric representation of a circumference in the space

The point p_i must not be on the axis otherwise the circumference degenerates into a point. To verify that this case does not occur, the following constraint must be satisfied:

$$|\delta^T r| < \|\delta\|$$

where $\delta = p_i - d$. If this relation is satisfied, it is possible to compute the position vector c of the centre of the circumference:

$$c = d + (\delta^T r)r$$

Knowing the position vector c, the radius ρ is computed:

$$\rho = ||p_i - c||$$

Moreover, the frame O'-x'y'z' in figure 4.10 is defined considering x' oriented along the direction of the vector (p_i – c), z' directed along vector r and y' to complete a right-handed frame. The rotation matrix R of this frame with respect to frame O-xyz is introduced:

$$R = [x'y'z']$$

These elements are the unit vectors of the frame O'-x'y'z' expressed in the base frame. Using all the element previously exposed, the time evolution of position and velocity are defined:

$$p(s) = c + R \begin{bmatrix} \rho cos\left(\frac{s}{p}\right) \\ \rho sin\left(\frac{s}{p}\right) \\ 0 \end{bmatrix}$$

$$\dot{p} = R \begin{bmatrix} -\dot{s} sin\left(\frac{s}{p}\right) \\ \dot{s} cos\left(\frac{s}{p}\right) \\ 0 \end{bmatrix}$$

These relations were implemented on MATLAB in discrete time domain. To show the obtained results, similarly to the previous case, two examples are proposed: a trajectory planning in 2-D and another in 3-D.

4.3.1 Circular trajectory planning in 2-D

The end-effector of the CNC machine can move along x-axis and z-axis; thus, the example exposed in this section provides a path planned in the xz frame. The parameters contained in table 4.3 were used to project the trajectory.

Using these parameters, the position vector of the centre of the circumference, the radius ρ, the time t_c and the acceleration $\ddot{p}_c$ were computed:

$$\text{c}=[5\ \text{-}8\ 5]^T$$

$$\rho = 2$$

$$t_c = 0.4292 \text{ s}$$

r	$[0\ 1\ 0]^T$
d	$[5\ -6\ 5]^T$
p_i	$[7\ -8\ 5]^T$
t_i	0 s
t_f	2 s
$\dot{p}_c$	8 cm/s

Table 4.3: Parameters employed in a 2-D circular trajectory planning

$$\ddot{p}_c = 18.6392\ cm/s^2$$

Considering that, in the case of a circular trajectory, $s_f = 2\pi\rho$, the arc length found is:

$$s(t) = \begin{cases} 9.3196t^2 & 0 \leq t \leq 0.4292 \\ 8(t - 0.2138) & 0.4292 < t \leq 1.5708 \\ 12.5664 - 9.3196(2 - t)^2 & 1.5708 < t \leq 2 \end{cases}$$

It was used with the matrix R and the radius ρ to compute p(t) and $\dot{p}(t)$. In figures 4.11 and 4.12 is shown the time evolution of x and y coordinates and the evolution of velocities along these axes. The planned circumference has centre in point (5,5), radius equal to 2 and the initial point of the path is (7,5). The velocities have not the trapezoidal profile because they are the velocities along x and z axes and the direction of motion changes incessantly during the path. In figure 4.13, it is shown the evolution of x coordinate with respect to z coordinate and the evolution of the velocity along the direction of motion. From the first graph, it is possible to notice that the path is circular as desired. In the second graph, the velocity has the correct behaviour and it reaches the desired cruise velocity $\dot{p}_c$ along the direction of motion.

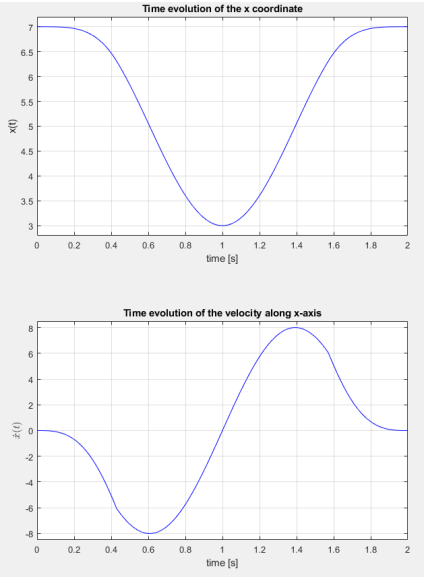

Figure 4.11: Time evolution of the x coordinate and velocity along x-axis in a 2-D circular trajectory planning

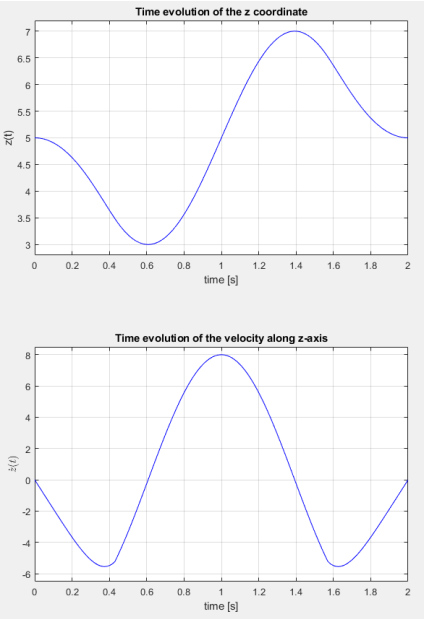

Figure 4.12: Time evolution of the z coordinate and velocity along z-axis in a 2-D circular trajectory planning

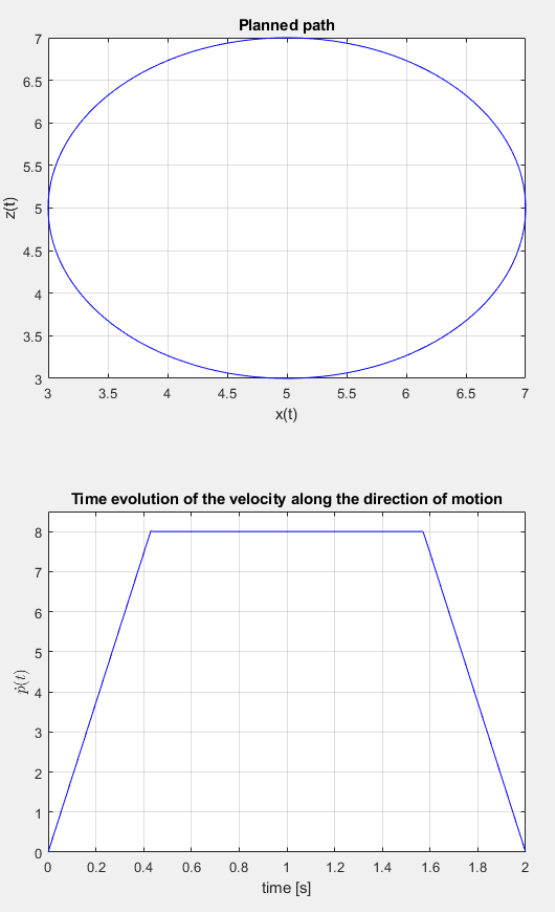

Figure 4.13: Planned path and time evolution of the velocity along the direction of motion in a 2-D circular trajectory planning

4.3.2 Circular trajectory planning in 3-D

In this section, an example about the planning of a circular trajectory in the space is exposed. The parameters contained in table 4.4 were used to project the trajectory. Using these parameters, the position vector of the centre of the circumference, the radius ρ, the time t_c and the acceleration $\ddot{p}_c$ were computed:

$$\text{c}=[5.6667\ 5.6667\ 5.6667]^T$$

$$\rho = 2.1602$$

$$t_c = 0.6427 \text{ s}$$

$$\ddot{p}_c = 15.5599\ cm/s^2$$

r	$[0.5774\ 0.5774\ 0.5774]^T$
d	$[1\ 1\ 1]^T$
p_i	$[6\ 7\ 4]^T$
t_i	0 s
t_f	2 s
$\dot{p}_c$	10 cm/s

Table 4.4: Parameters employed in a 3-D circular trajectory planning

The arc length found is:

$$s(t) = \begin{cases} 7.78t^2 & 0 \leq t \leq 0.6427 \\ 10(t - 0.3213) & 0.6427 < t \leq 1.3573 \\ 13.5732 - 7.78(2 - t)^2 & 1.3573 < t \leq 2 \end{cases}$$

It was used with the matrix R and the radius ρ to compute p(t) and $\dot{p}(t)$.

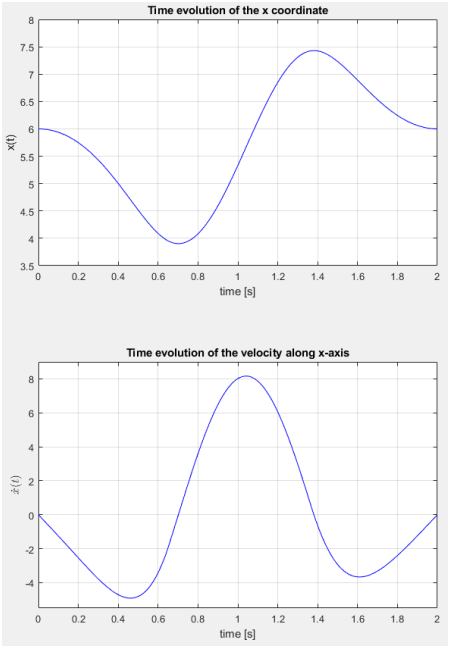

Figure 4.14: Time evolution of the x coordinate and velocity along x-axis in a 3-D circular trajectory planning

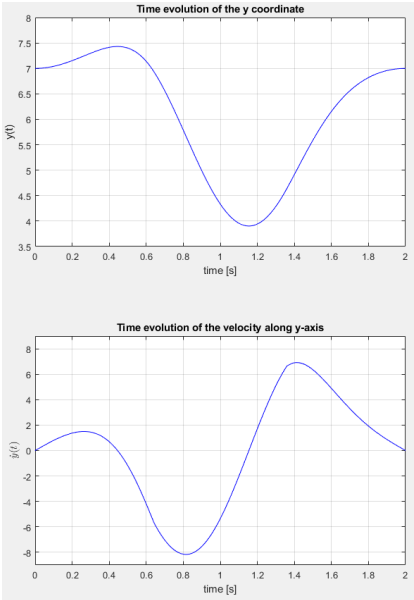

Figure 4.15: Time evolution of the y coordinate and velocity along y-axis in a 3-D circular trajectory planning

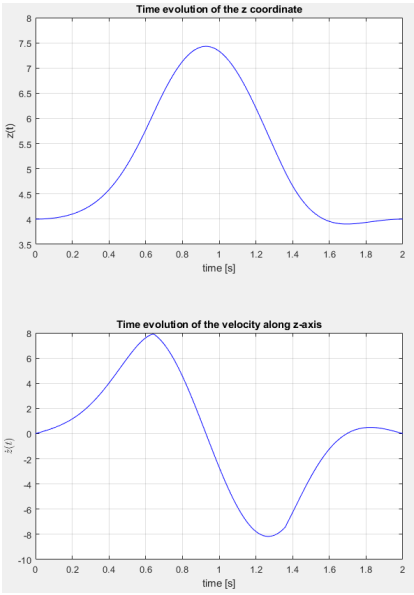

Figure 4.16: Time evolution of the z coordinate and velocity along z-axis in a 3-D circular trajectory planning

In figures 4.14, 4.15 and 4.16 is shown the time evolution of x, y and z coordinates and the evolution of velocities along these axes. In figure 4.17, it is shown the planned path and the evolution of the velocity along the direction of motion. From the first graph, it is possible to notice that the path has the desired shape, its centre (red point) is the set one and it pass for the point p_i (black point). In the second graph, the velocity has the correct behaviour and it reaches the set cruise velocity $\dot{p}_c$ along the direction of motion.

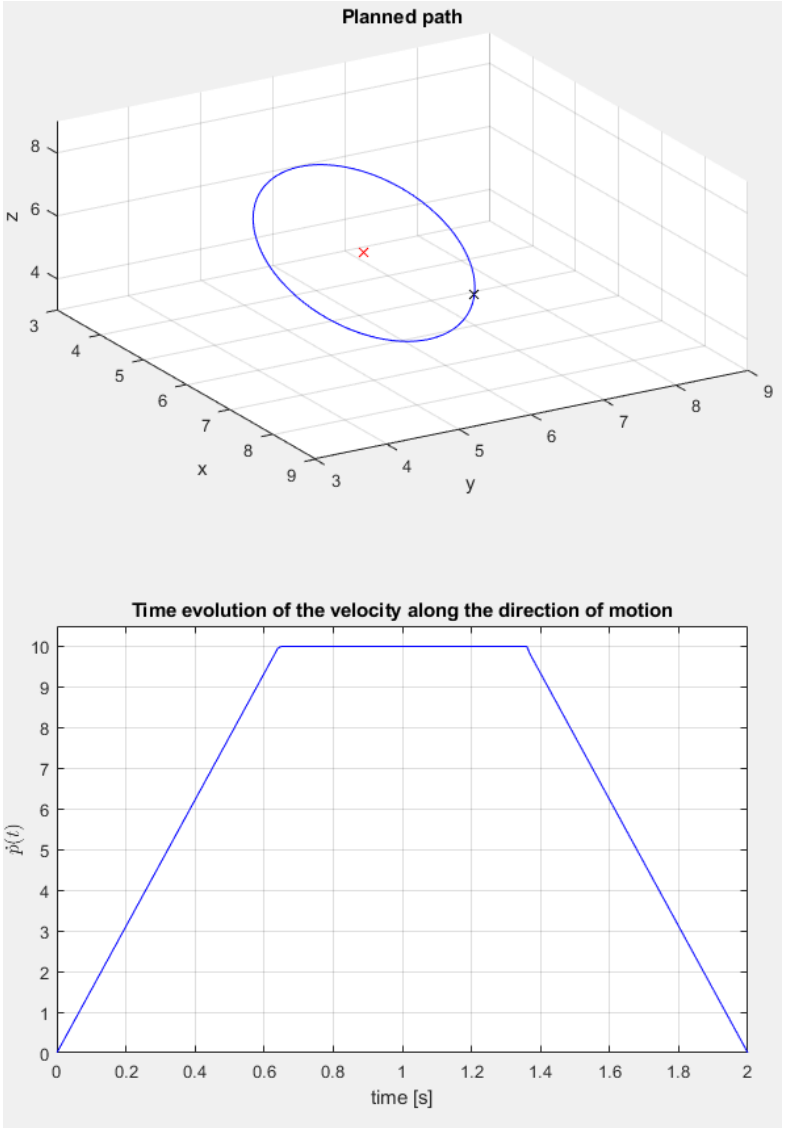

Figure 4.17: Planned path and velocity along the direction of motion in a 3-D circular trajectory planning

4.4 Test of the CNC machine model

The 2-D trajectories described in this chapter have to be used as input for the CNC machine model, in order to obtain a simulation with a plausible behaviour. Before moving on the development of the algorithm able to identify possible intersection between the end-effector tool and the workpiece, obviously, some tests about the response of the model to these inputs were conducted. In this section, it is exposed the test in which the provided input is the trajectory described in section 4.2.1. However, also the test with circular trajectory gives positive results.
The following figures illustrate the results obtained from this test. Figures 4.18 and 4.19 shows how the desired coordinates of the end-effector frame x_d and z_d and the effective ones x_e and z_e vary in time. It is possible to notice that the model is able to follow the input signal behaviour. Figures 4.20 and 4.21 contains a comparison between the time evolution of the desired velocities along axes ($\dot{x}_d$ and $\dot{z}_d$) and the effective ones ($\dot{x}_e$ $\dot{z}_e$). Also in this case, the model is able to follow the planned trajectory. In figure 4.22, the velocities along the direction of motion $\dot{p}_d$ and $\dot{p}_e$ are compared. Finally, figure 4.23 contains in its first graph the evolution of position errors e_x (blue) and e_z (red). Instead, the second graph shows the errors $e_{\dot{x}}$ (blue) and $e_{\dot{z}}$ (red) that involves the velocities along the x and z axes.

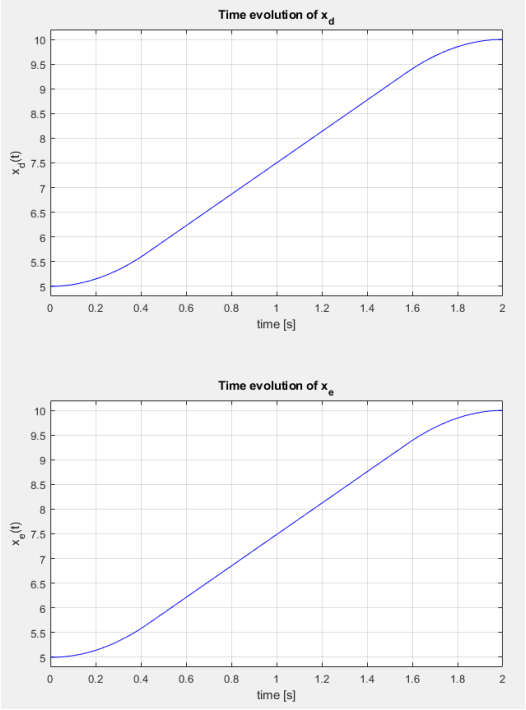

Figure 4.18: Time evolution of the desired and the effective x coordinate

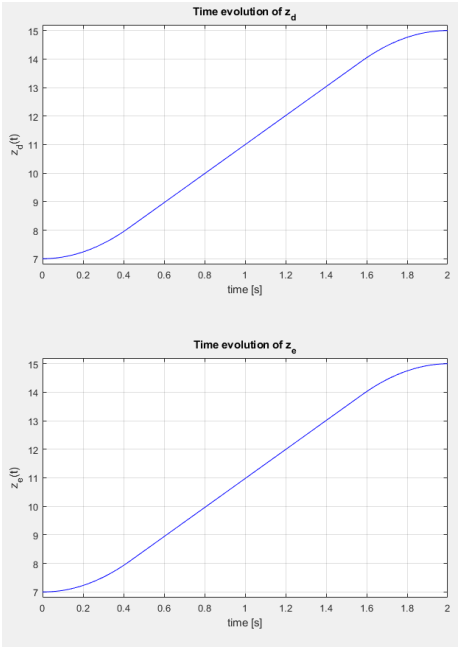

Figure 4.19: Time evolution of the desired and the effective z coordinate

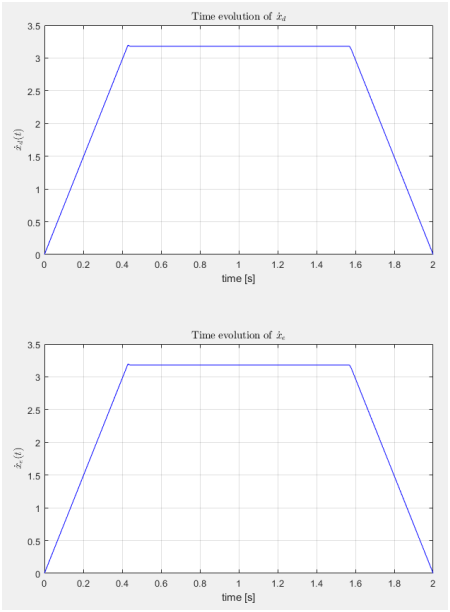

Figure 4.20: Time evolution of the desired and the effective velocity along x-axis

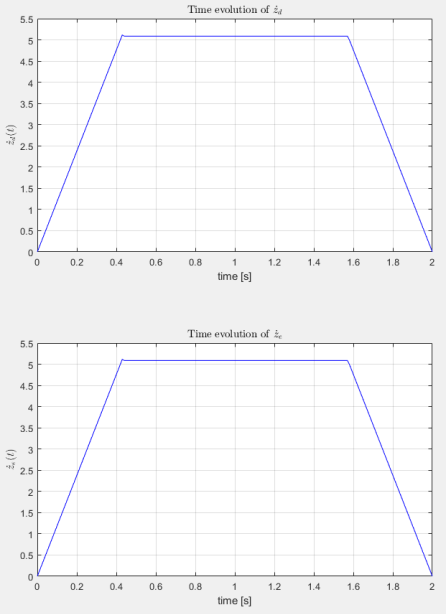

Figure 4.21: Time evolution of the desired and the effective velocity along z-axis

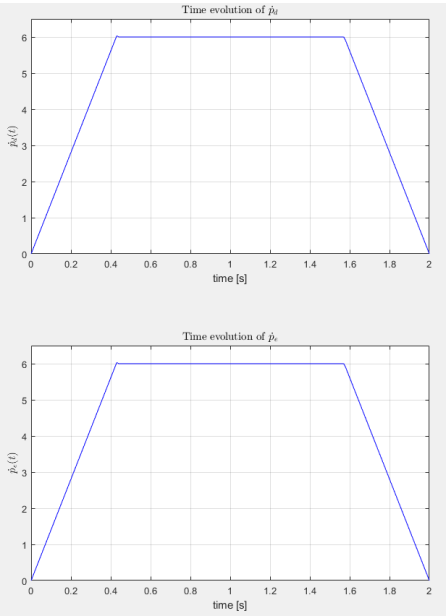

Figure 4.22: Time evolution of the desired and the effective velocity along the direction of motion

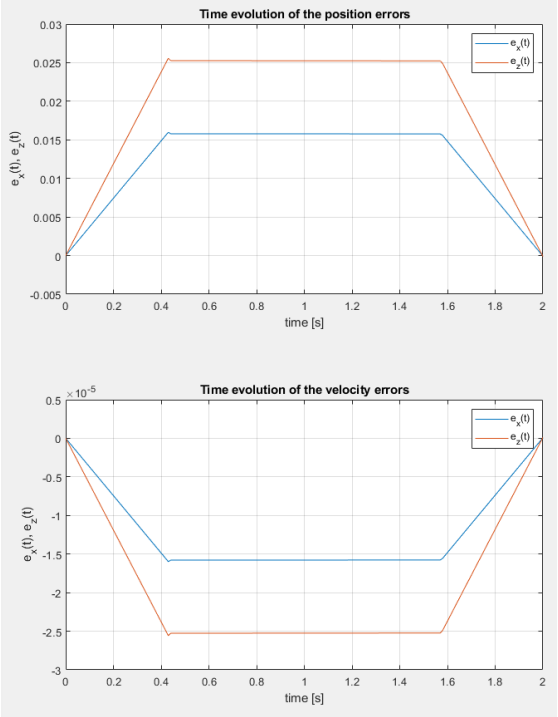

Figure 4.23: Time evolution of the position and velocity errors

The result of this test is positive: the model is able to follow the desired behaviour with a small error. The objective of the first part of this thesis work was to create a simple model able to reproduce the movements of the CNC machine in order to develop the algorithm to identify the interaction. For this reason, the algorithm employed to create the model was the inverse kinematic one. In future, the error could be reduced by implementing a better control strategy to simulate the CNC machine behaviour.

Chapter 5

Interaction identification

Once the model was completed, being able to simulate the movements of the CNC machine, it was possible to know the position of the frame attached to the end-effector tool. Using this position, the end-effector was approximated to a point cloud and its interaction with the plane, which represents the work surface of the piece, was analysed. From this operation, the points of the end-effector that are in contact with the workpiece are identified. In the second part of this project which will be covered by my course mate Emanuele Pansica, these points will be used in a mechanical analysis in order to produce virtual data.

5.1 End-effector tool approximation

The end effector tool was approximated to a point cloud with cylindrical shape. It is not a plausible form but, in the future, it will be possible to modify it and to adopt a more realistic shape. The approximation is obtained by creating numerous circumferences with a constant distance between them, starting from the position of the end-effector frame with respect to the base frame. It was located at the base of the end-effector tool and its origin coincides with the centre of the tool. The x and z coordinates of the frame vary as the manipulator configuration changes; on the other hand, the y-coordinate is fixed, it does not depend on the joint variable. From the direct kinematics analysis, in chapter 3, it is known that the relation between end-effector position and joint variables is the following one:

$$p_e(t) = \begin{bmatrix} x_e(t) \\ y_e(t) \\ z_e(t) \end{bmatrix} = \begin{bmatrix} q_2(t) \\ -\overline{d} \\ q_1(t) \end{bmatrix}$$

The circumferences were generated using this position as centre. Thus, as the manipulator configuration changes, the circumferences position also changes. The position of the points that compose these circumferences was obtained using the following relation:

$$p_i(t) = \begin{bmatrix} x_i(t) \\ y_i(t) \\ z_i(t) \end{bmatrix} = \begin{bmatrix} x_e(t) + r\cos(\theta + \phi) \\ y_e(t) - k \\ z_e(t) + r\sin(\theta + \phi) \end{bmatrix}$$

where:

- p_i is the position of the generic point i and x_i, y_i and z_i are its coordinate;
- r is the radius of the cylinder that approximate the cutting tool;
- θ is an angle that varies in the interval [0,2π];
- ϕ is an angle that depends on the considered time instant and the rotation velocity of the end-effector tool;
- k determines the position of the circumference in the y axis, it varies between zero and the cutting tool length.

This relation was implemented on MATLAB using two for cycles. One to vary the angle θ, in order to define a complete circumference and the other to vary k, defining numerous identical circumferences with the same centre equally distributed along the y-axis. Obviously, this relation was implemented in a discrete manner. Actually, the algorithm computes the angle ϕ considering the rotational velocity of the cutting tool as constant. However, during the cutting work, its value experiences some variations. For this reason, when the mechanical analysis will be completed, it will be possible to know the value of the rotational velocity in each time instant and to use it to compute the angle ϕ.
The figure 5.1 shows the approximated end-effector tool at different time instant, the red line was added in order to show the rotation of the tool.

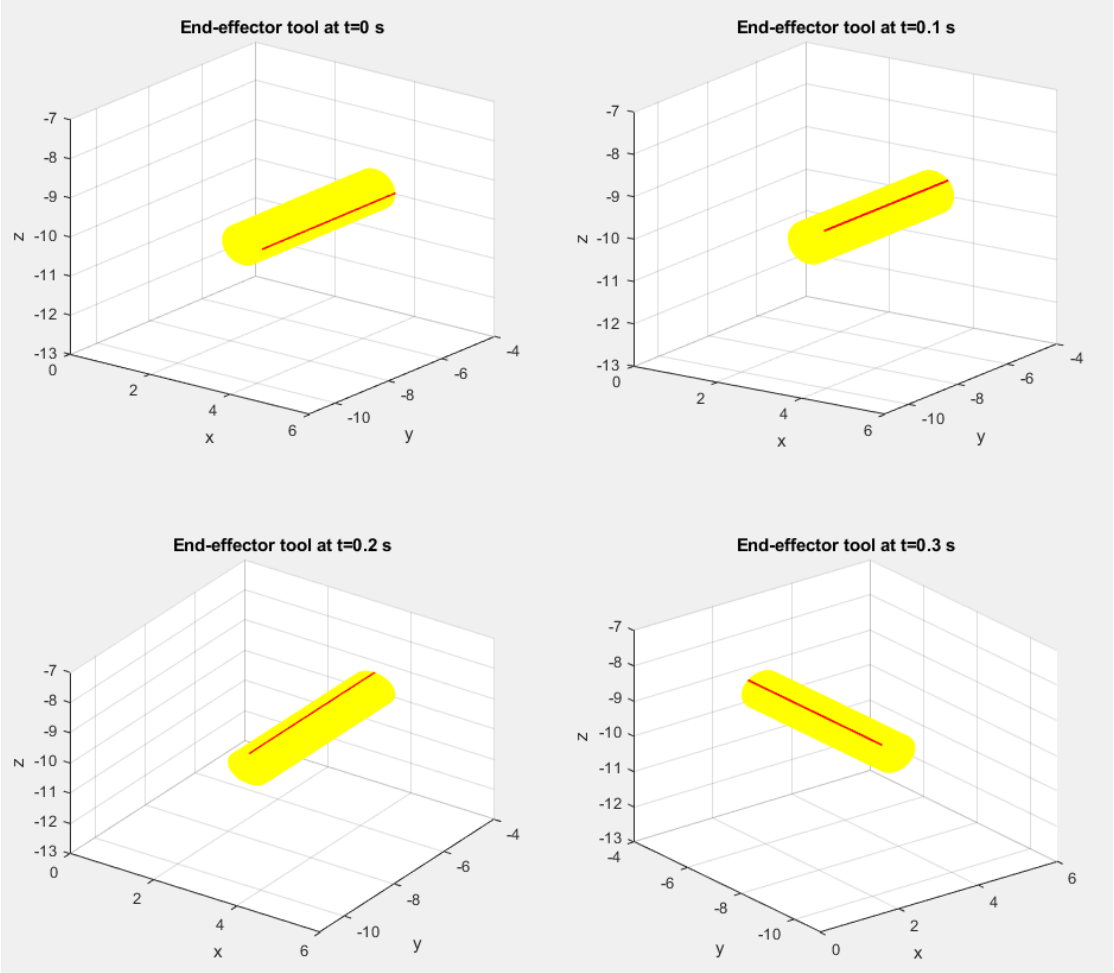

Figure 5.1: Approximated end-effector tool at different time instant

5.2 Identification of the contact points

The next step was the development of an algorithm able to identify the points of the end-effector tool that are in contact with the workpiece during the simulation of cutting operations at each time instant. Specifically, the points that are under the work surface considered in chapter 3. This operation is made in two steps:

- Identify the points that could have a contact with the plane considering only the x and z axes, they were termed possible contact points;
- Identify the contact points starting from the possible contact ones and considering the y coordinate of the plane and the end-effector points.

5.2.1 Possible contact points

In this part, the 3-D problem is reconducted to a 2-D one, considering only the x and z axes. As first step, the four vertices of the plane are computed using the transformation matrix exposed in chapter 3. Considering the following equation of a straight line:

$$x = mz + q$$

The x and z coordinates of the vertices are used to compute the coefficients m and q of the lines that compose the sides of the plane using the following formulas:

$$m = \frac{x_1 - x_2}{z_1 - z_2}$$

$$q = \frac{z_1 x_2 - z_2 x_1}{z_1 - z_2}$$

Afterward, the end-effector points are considered. Being composed by numerous identical circumferences equally distributed along the y-axis and considering the 2-D case, it is possible to consider the points that compose only one circumference because the results are the same for all of them. For each point of the circumference, it is computed the distance d from each side of the 2-D plane, using the following formula:

$$d = \frac{|x_p - (mz_p + q)|}{\sqrt{1 + m^2}}$$

where x_p and z_p are the coordinates of the circumference point. Then, for each point, four value are obtained and they are compared with two thresholds in order to identify the possible contact points.
In the described procedure, some problems arise when there is not a rotation around the x-axis. In this case, the vertices have the same z coordinate in pairs, thus two sides are parallel to the x-axis and the coefficients m and q are not computable. For them, the distance from the end-effector points is computed using the following formula:

$$d = z_p - z_v$$

where z_v is the z coordinate of a vertices.
The exposed procedure was developed on MATLAB. Figure 5.2 shows the code used to obtain the coefficient m and q and the two thresholds. The elements in the first column of the matrix “rette” are the coefficients m for each side of the plane and the elements in the second one are the coefficients q. The terms in the matrix

```
rette=[((p_m(1,1)-p_m(2,1))/(p_m(1,3)-p_m(2,3))) ((p_m(1,3)*p_m(2,1)-p_m(2,3)*p_m(1,1))/(p_m(1,3)-p_m(2,3)));
    ((p_m(2,1)-p_m(3,1))/(p_m(2,3)-p_m(3,3))) ((p_m(2,3)*p_m(3,1)-p_m(3,3)*p_m(2,1))/(p_m(2,3)-p_m(3,3)));
    ((p_m(3,1)-p_m(4,1))/(p_m(3,3)-p_m(4,3))) ((p_m(3,3)*p_m(4,1)-p_m(4,3)*p_m(3,1))/(p_m(3,3)-p_m(4,3)));
    ((p_m(1,1)-p_m(4,1))/(p_m(1,3)-p_m(4,3))) ((p_m(1,3)*p_m(4,1)-p_m(4,3)*p_m(1,1))/(p_m(1,3)-p_m(4,3)))];

max_dist_x0=abs(p_m(4,1)-(rette(1,1)*p_m(4,3)+rette(1,2)))/sqrt(1+(rette(1,1))^2);
max_dist_z0=abs(p_m(1,1)-(rette(2,1)*p_m(1,3)+rette(2,2)))/sqrt(1+(rette(2,1))^2);
```

Figure 5.2: MATLAB code used to compute the coefficients m and q and the thresholds

"p_m" are the x and z coordinates of the four vertices. To conclude, the terms "max_dist_x0" and "max_dist_z0" are the two thresholds.

In figure 5.3 is shown the code used to identify the possible contact points. The first if function is used to detect whether the condition is the normal one or the one without rotation around the x-axis. Successively, the for loops in each part of the if function are used to consider each point of the circumference. For each point the distance from the sides of the plane is computed in vector "distan". Finally, these values are compared with the thresholds using an if function in order to identify the possible contact points.

```
if (p_m(1,3)==p_m(4,3) || p_m(2,3)==p_m(3,3))
    for i=1:length(angle)
        distan(i,:)=[abs(posizione_punta(i,1)-(rette(1,1)*posizione_punta(i,3)+rette(1,2)))/sqrt(1+(rette(1,1))^2)
            abs(posizione_punta(i,3)-p_m(2,3))
            abs(posizione_punta(i,1)-(rette(3,1)*posizione_punta(i,3)+rette(3,2)))/sqrt(1+(rette(3,1))^2)
            abs(posizione_punta(i,3)-p_m(4,3))];
        if(distan(i,1)<=max_dist_x0 && distan(i,2)<=max_dist_z0 && distan(i,3)<=max_dist_x0 && distan(i,4)<=max_dist_z0)
            k=k+1;
            possible_contact(k,:)=[posizione_punta(i,1) posizione_punta(i,3)];
        end
    end
else
    for i=1:length(angle)
          distan(i,:)=[abs(posizione_punta(i,1)-(rette(1,1)*posizione_punta(i,3)+rette(1,2)))/sqrt(1+(rette(1,1))^2)
              abs(posizione_punta(i,1)-(rette(2,1)*posizione_punta(i,3)+rette(2,2)))/sqrt(1+(rette(2,1))^2)
              abs(posizione_punta(i,1)-(rette(3,1)*posizione_punta(i,3)+rette(3,2)))/sqrt(1+(rette(3,1))^2)
              abs(posizione_punta(i,1)-(rette(4,1)*posizione_punta(i,3)+rette(4,2)))/sqrt(1+(rette(4,1))^2)];
          if(distan(i,1)<=max_dist_x0 && distan(i,2)<=max_dist_z0 && distan(i,3)<=max_dist_x0 && distan(i,4)<=max_dist_z0)
            k=k+1;
            possible_contact(k,:)=[posizione_punta(i,1) posizione_punta(i,3)];
          end
    end
end
```

Figure 5.3: MATLAB code used to identify the possible contact points

In figure 5.4 is shown a graphical result of the algorithm application in the normal case. The contact surface is blue, the possible contact points are red and the end-effector points that could not be in contact with the workpiece are in yellow. Figure 5.5 illustrates the result when there is not a rotation around the x-axis.

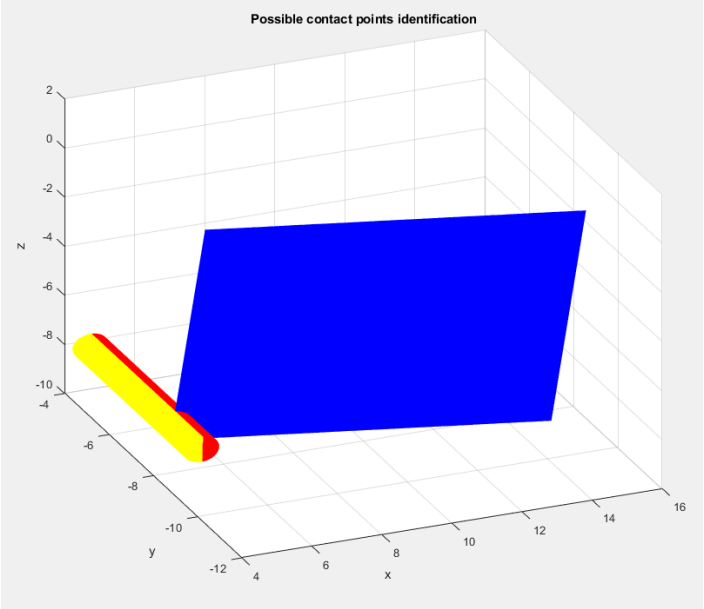

Figure 5.4: Possible contact points identification in normal condition

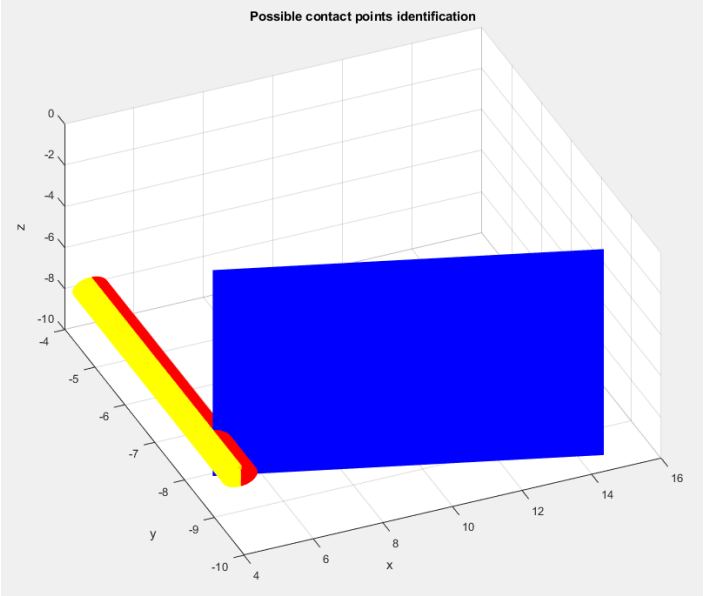

Figure 5.5: Possible contact points identification when the contact surface was not subject to a rotation around the x-axis

5.2.2 Contact points

The second part of the algorithm uses the y-coordinate of the possible contact points to identify if the end-effector tool is in contact with the working surface. In particular, for each possible contact point is considered each y-coordinate of the circumferences which approximate the end-effector tool and it is compared with the y-coordinate of the plane in the x and z coordinates in that point. Thus, it is necessary to compute the equation of the plane using three vertices. The generic equation of a plane is:

$$ax + by + cz + d = 0$$

Termed the vertices as A, B and C, the vectors AB and AC are computed:

$$\vec{AB} = \vec{B} - \vec{A}$$

$$\vec{AC} = \vec{C} - \vec{A}$$

The cross product between these vectors is calculated and it is used to identify the coefficient a, b and c of the plane equation. Lastly, the parameter d is obtained by imposing that the plane passes for the point A:

$$d = -ax_A - by_A - cz_A$$

Knowing the plane equation, it is possible to obtain the y-coordinate for each couple of x and z coordinates. Thus, considering the generic possible contact point i which has as x and z coordinates (x_i, z_i), substituting them in the plane equation the y-coordinate y_i of the plane in that point is obtained. Afterward, this value is compared with the different y-coordinates of the circumferences that compose the end-effector tool, in order to identify the contact points.
This part of the algorithm was implemented on Simulink with the previous part using the MATLAB function block. In figure 5.6 is shown the code used to implement the contact point identification. The first if function is employed to verify if possible contact points have been detected. The first for loop is utilized to consider each possible contact point; then, the y-coordinates of the plane are computed and compared with those of the circumferences using the second for loop. In this loop, the points that are under the working surface ("contact") and the points that are directly in contact with the plane ("exact_contact") are identified. The last points could be employed to know the depth of penetration. The vectors "contact" and "exact_contact" have variable size because they could be empty when there is not contact between the end-effector and the plane or they could have a variable number of elements depending on the end-effector position and on the depth of penetration. To use variable size vectors on Simulink, it is necessary to use a specific method. For this reason, the last part of the code was added.

```
if(k>0)
    for i=1:k
        y_plane = (-(a*possible_contact(i,1)+c*possible_contact(i,2)+d))/b;
        for j=0:0.05:5
            distance_plane_point=(-d_bar-j)-y_plane;
            if distance_plane_point<0
                l=l+1;
                flag_contatto=1;
                if abs (distance_plane_point)<0.05
                    m=m+1;
                    exact_contact_1(m,:)=[possible_contact(i,1) y_plane possible_contact(i,2)];
                    contact_1(l,:)=[possible_contact(i,1) y_plane possible_contact(i,2)];
                    l=l+1;
                    contact_1(l,:)=[possible_contact(i,1) -d_bar-j possible_contact(i,2)];
                else
                    contact_1(l,:)=[possible_contact(i,1) -d_bar-j possible_contact(i,2)];
                end
            end
        end
    end
    exact_contact=exact_contact_1(1:m,:);
    contact=contact_1(1:l,:);
else
    exact_contact=exact_contact_1;
    contact=contact_1;
end
```

Figure 5.6: MATLAB code used to identify the contact points

The following figures shows the results. In figures 5.7 and 5.8, it is considered the case when the end-effector is not completely inside the workpiece. In the second one, it is shown the interaction from the x-y plane point of view. It is possible to see that the contact points (red) are perfectly identified. In figures 5.9 and 5.10 is shown the case when the end-effector is completely inside the working plane. The second one shows the x-y plane viewpoint.

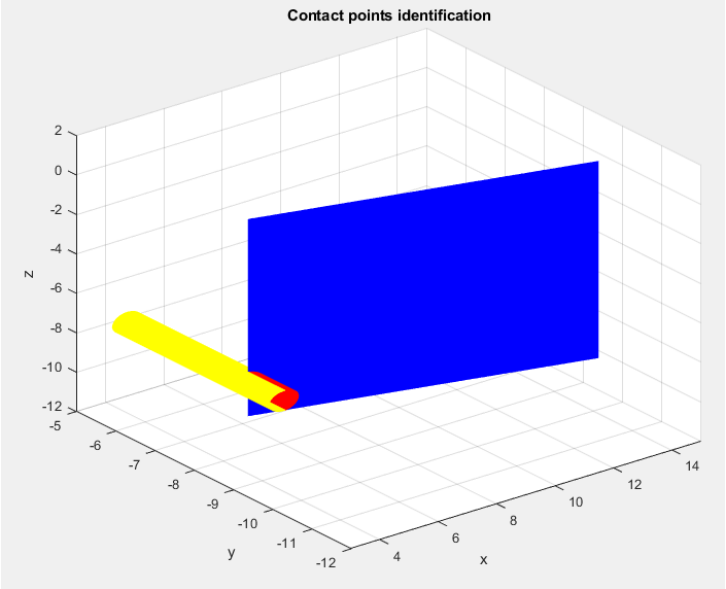

Figure 5.7: Contact points identification when the end-effector is not totally inside the working surface

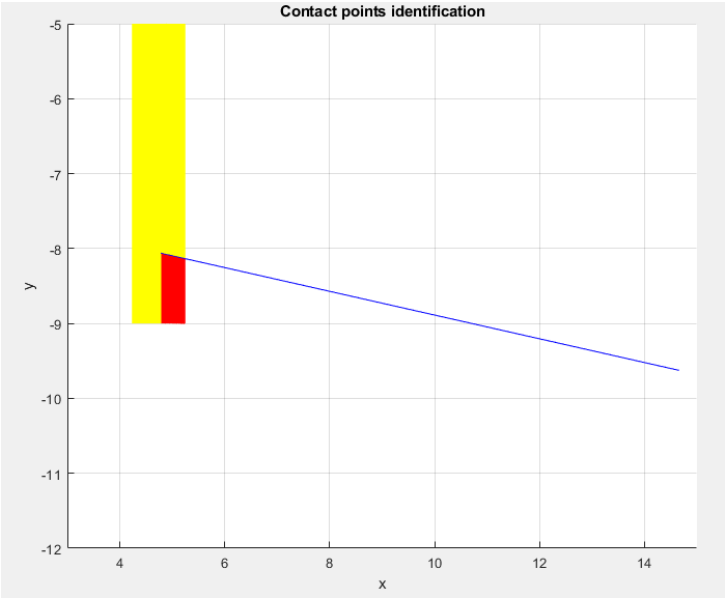

Figure 5.8: Contact points identification when the end-effector is not totally inside the working surface from the x-y plane viewpoint

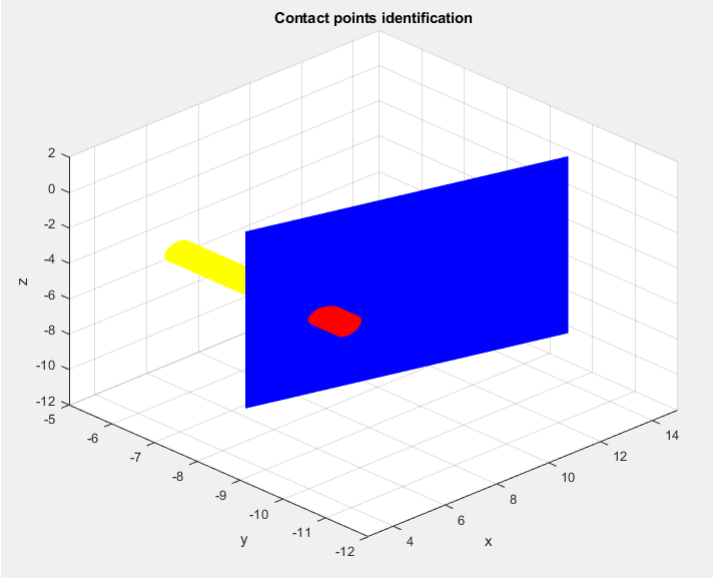

Figure 5.9: Contact point identification when the end-effector is totally inside the working surface

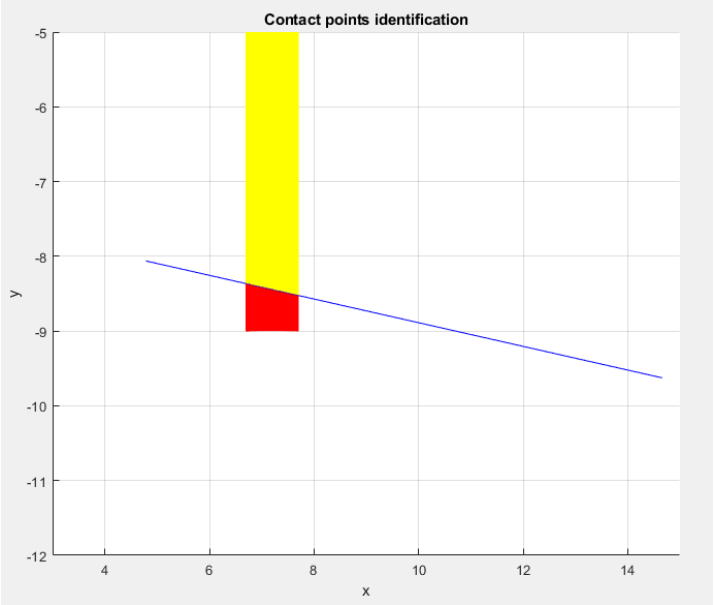

Figure 5.10: Contact point identification when the end-effector is totally inside the working surface from the x-y plane viewpoint

5.3 Final test

The final part of this thesis work was to assemble all the different parts exposed in the previous chapters with the contact points identification algorithm. The goal was to obtain a simulation in which the CNC machine model follows the input trajectory and, during its movements, it is identified if the end-effector tool is in contact with the working plane and, if this occurs, to detect the contact points. This was made on Simulink, in figure 5.11 is shown the structure of the Simulink implementation of the complete system. It is composed by the subsystem "CNC model" (light blue) which contains the implementation of the inverse kinematics algorithm exposed in chapter 3. The inputs of this block are the desired position and velocity obtained by the trajectory planning; the output is the position of the end-effector frame. It is used by the MATLAB function "End-effector tool approximation" (yellow) to generate the circumferences that creates the approximation of the end-effector tool. The MATLAB function "Plane position" (blue) computes the position of the working surface vertices after the rotations and the translation imposed by the second part of the CNC machine. These positions are used with the positions of the end-effector tool points by the MATLAB function "Contact points identification" (green) to detect a possible interaction between the end-effector tool and the working plane and, eventually, to identify the contact points. One of the outputs of this block is the variable "flag_contatto", it is equal to zero when there is no interaction and, in the other case, its value is set to one. The last MATLAB function "Print 3d" (orange) produces graphical representations of the working process, it is useless for the objective of the simulator because only the contact point positions are relevant, being successively used in the mechanical analysis, however, in this part of the project, this block was employed to check the correct behaviour of the system and to observe the final results.

Figure 5.12 and 5.13 shows the results of a simulation of the working process. They contain the graphical representations of the process at certain time instants that were chosen to illustrate the different conditions of the activity. In particular, the time instants in which the end-effector is approaching the working surface were chosen, to show that the algorithm works correctly.

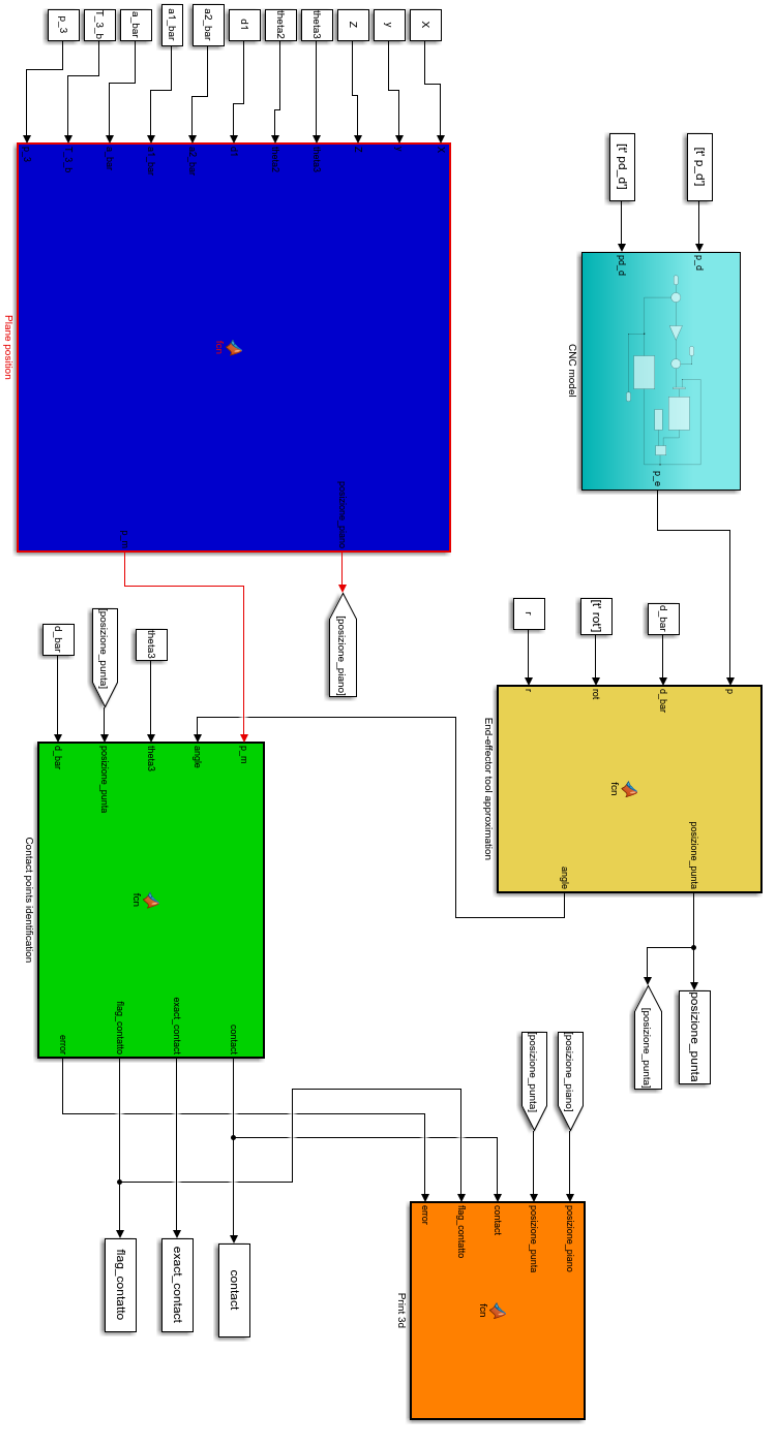

Figure 5.11: Simulink implementation of the complete system

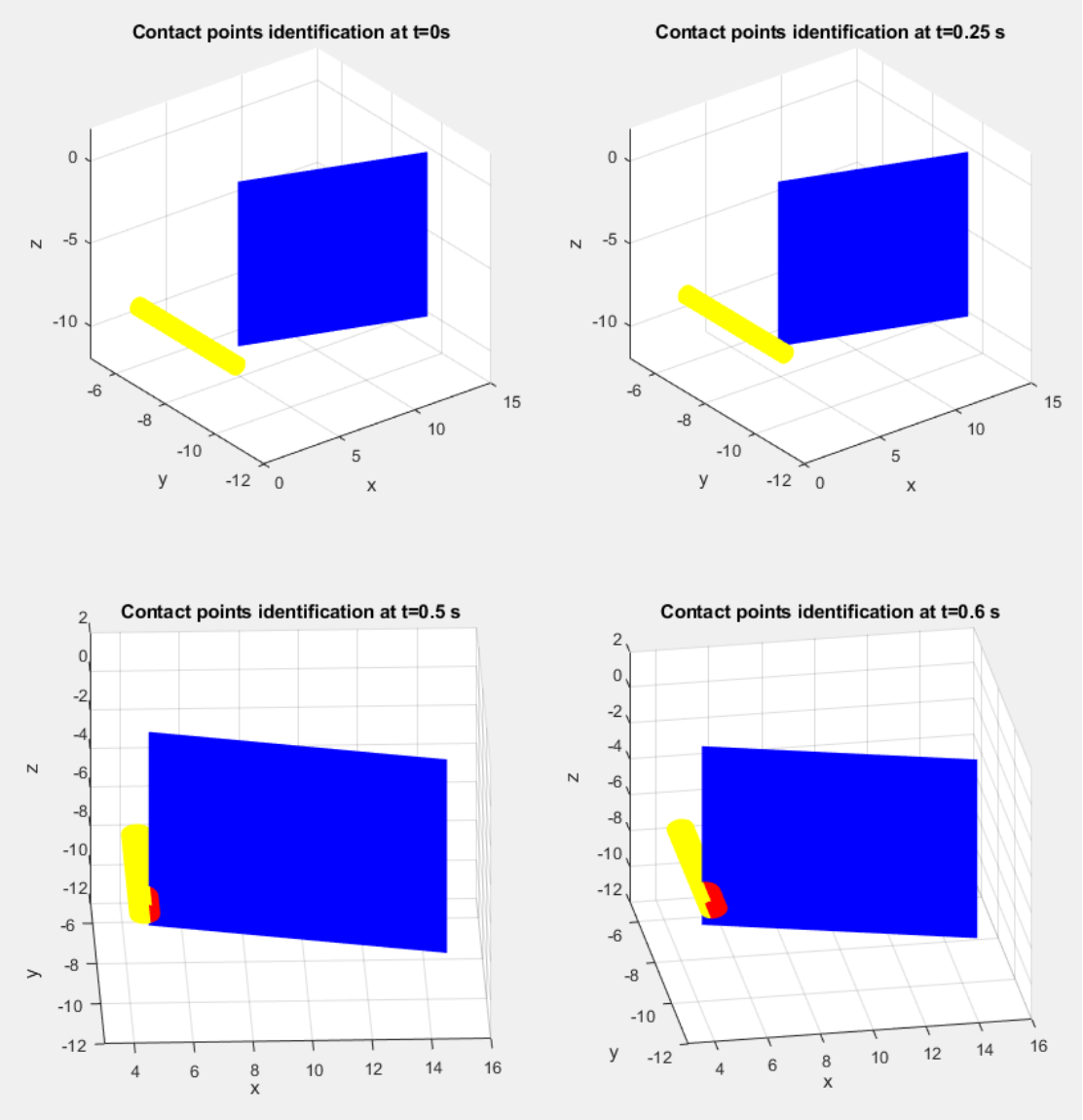

Figure 5.12: First part of the graphical result of the cutting process simulation

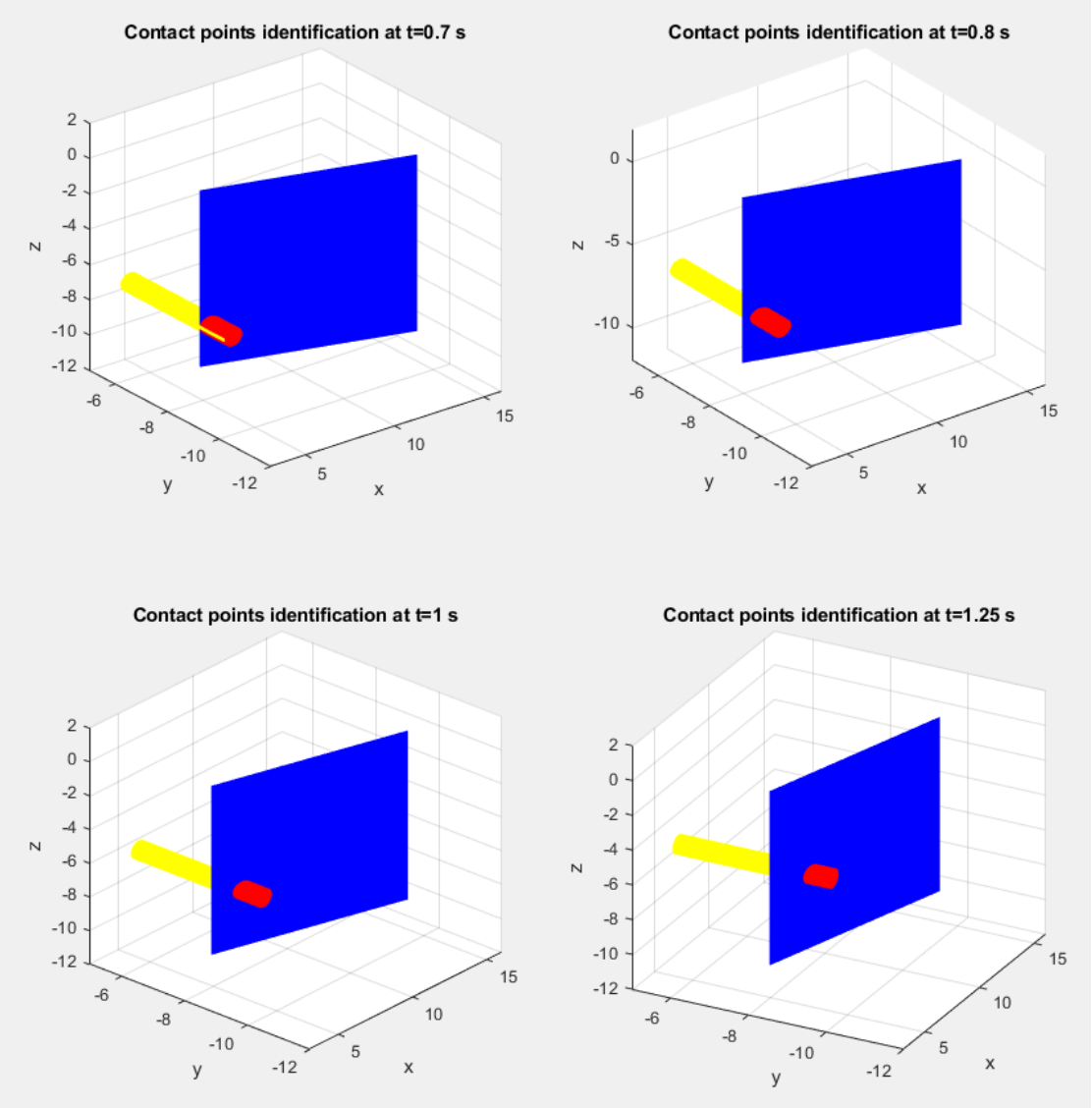

Figure 5.13: Second part of the graphical result of the cutting process simulation

Chapter 6

Conclusions and future works

In this thesis the initial phase of the development of a CNC machine simulation was analysed. The main purpose of this simulation is to produce artificial data to be employed in the study of a suitable SOH prediction technique for the end-effector tool. The applied approach is based on the Finite Element Method, in order to analyse the interaction between tool and workpiece in detail. In fact, by using this method it is possible to examine each part of the cutting tool and to consider the parameters involved in the working process which are not constant on its surface. The main activities performed in this thesis work are:

- the creation of a CNC machine simulation able to reproduce its movements, starting from the direct kinematics and the differential kinematics analyses;
- the development of an algorithm able to plan a rectilinear trajectory and another to plan a circular one. Based on the structure of the CNC machine considered, these algorithms are used in 2-D trajectory planning; however, they work properly also in the 3-D case;
- the development of an algorithm which identifies the contact points between the end-effector tool and the workpiece during the working process;
- the combination of the previous parts in order to create a simulation in which the CNC machine model moves following the desired trajectory and, at each instant of time, the contact points are identified.

This simulation was tested, obtaining positive results. Thus, the objectives set were reached and it is possible to proceed with the second part of this project.

6.1 Future works

The second part of the simulation will use the obtained contact points in a mechanical analysis in order to produce artificial data. Once identified the points in which the end-effector tool and the workpiece interact, it will be possible to analyse the contact surface considering in a microscopical viewpoint the different phenomena that occur, as the temperature that is not constant in the surface but varies depending on the considered zone. When this part is completed, the simulation could be improved by developing an algorithm able to plan more complex trajectories or by using a more realistic approximation of the end-effector tool.

When the simulation is able to produce realistic data, they will be used in the future works of the MOREPRO project to develop a suitable method able to predict the wear condition of the cutting tool.

Bibliography

[1] Hashem M Hashemian. «State-of-the-art predictive maintenance techniques». In: *IEEE Transactions on Instrumentation and measurement* 60.1 (2010), pp. 226–236 (cit. on p. 2).

[2] Rafael Gouriveau, Kamal Medjaher, and Noureddine Zerhouni. *From prognostics and health systems management to predictive maintenance 1: Monitoring and prognostics*. John Wiley & Sons, 2016 (cit. on p. 7).

[3] Aleksandra Marjanović, Goran Kvaščev, Predrag Tadić, and Željko Đurović. «Applications of predictive maintenance techniques in industrial systems». In: *Serbian Journal of Electrical Engineering* 8.3 (2011), pp. 263–279 (cit. on pp. 8, 9, 11, 12).

[4] WorldQuant Perspectives. *Reshaping the world with Fuzzy Logic*. April 2018 (cit. on p. 12).

[5] Alessandro Paolo Daga and Luigi Garibaldi. «Machine vibration monitoring for diagnostics through hypothesis testing». In: *Information* 10.6 (2019), p. 204 (cit. on p. 13).

[6] Luca Cecere. «End-effector tools SoH prediction: parameter identification and reliability estimation.» PhD thesis. Politecnico di Torino, 2021 (cit. on p. 14).

[7] Michele Pinto. «End-effector tools wear prediction: a multimodel approach». PhD thesis. Politecnico di Torino, 2021 (cit. on p. 14).

[8] Antonia Verde. «End-effector tools wear prediction: machine and interaction modeling, system identification based on the EKF approach.» PhD thesis. Politecnico di Torino, 2021 (cit. on p. 14).

[9] Sunday J Ojolo, Olumuwiya Agunsoye, Oluwole Adesina, and Gbeminiyi M Sobamowo. «Cutting Force and Friction Modelling in High Speed End-Milling». In: *International Manufacturing Science and Engineering Conference*. Vol. 56826. American Society of Mechanical Engineers. 2015, V001T02A029 (cit. on p. 16).

[10] Abdil Kus, Yahya Isik, M Cemal Cakir, Salih Coşkun, and Kadir Özdemir. «Thermocouple and infrared sensor-based measurement of temperature distribution in metal cutting». In: *Sensors* 15.1 (2015), pp. 1274–1291 (cit. on p. 17).

[11] Jaroslav Mackerle. «Finite element analysis and simulation of machining: an addendum: A bibliography (1996–2002)». In: *International Journal of Machine Tools and Manufacture* 43.1 (2003), pp. 103–114 (cit. on p. 18).

[12] Zhao Haitao, Yang Jianguo, and Shen Jinhua. «Simulation of thermal behavior of a CNC machine tool spindle». In: *International Journal of Machine Tools and Manufacture* 47.6 (2007), pp. 1003–1010 (cit. on p. 18).

[13] Mikell P. Groover. *Automation, Production Systems, and Computer-Integrated Manufacturing*. Pearson, 2015 (cit. on p. 20).

[14] Luigi Villani Bruno Siciliano Lorenzo Sciavicco and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009 (cit. on pp. 22, 26, 29, 40, 42, 51).