

**New concepts in automation and robotic
technology for surface engineering**

Master Thesis

by

Juan Antonio García Marín
from Murcia, Spain

at

**Institute for Manufacturing Technologies of Ceramics
Components and Composites- IMTCCC**

University of Stuttgart

Stuttgart, July 2010

ACKNOWLEDGMENT

My thanks to everyone who have made possible the realization of this project. Although I cannot name all of them I want to specifically recognize the value of some of them:

To Prof. Dr. Rainer Gadow for giving me this opportunity to realize my final thesis at the Institute for Manufacturing Technologies of Ceramic Components and Composites (IMTCCC; University of Stuttgart), as well as Dr. Andreas Killinger and Dipl.-Ing. Miriam Floristan for their mentoring and support throughout the entire work.

To D. José Conde del Teso and D. Antonio Guerrero as my teachers in the University Polytechnic of Cartagena.

To the IFKB team, without whose help and advice the success of this Final Thesis would have not been possible.

To my family, for their unconditional support and motivation.

To my friends, and classmates for their encouragement.

And to all of them, who I forgot to mention.

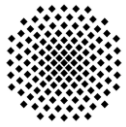
Thanks to all of them, I was able to achieve this aim in my life.

Stuttgart, 15th July 2010



Page Index

CHAPTER 0: MOTIVATION	1
CHAPTER 1: THERMAL SPRAYING AUTOMATION	2
1.1 Principles of Thermal Spraying	2
1.2 Thermal spraying processes	3
1.3 Use of robots in Thermal Spraying	7
CHAPTER 2: INDUSTRIAL ROBOTS	10
2.1 Definition of Industrial Robot	11
2.2 Classification of robots	11
2.3 Basic Components of Industrial Robots	13
2.4 Coordinate Systems	16
2.5 Robots Kinematics. Forward and Inverse Kinematics	18
CHAPTER 3: ROBOTS PROGRAMMING	20
3.1 Levels of programming	20
3.2 Methods of programming	21
CHAPTER 4: DESIGN OF TRAJECTORIES	24
4.1 Steps to create a program	24
4.2 Description of the applications	27
4.2.1 Meander A	28
4.2.2 Meander B	30
4.2.3 Meander C	31
4.2.4 Triangular (Mirror)	32
4.2.5 Variable Meander	34
4.3 Translation to Robot's language	35
4.3.1 V+ Language	35
4.3.2 Karel Language	37
4.4 Equipments	38



CHAPTER 5: OFF-LINE PROGRAMMING	41
5.1 Criteria to evaluate simulation software	41
5.2 Simulation software in the market	44
5.3 Roboguide – Fanuc	51
5.3.1 Main features	51
5.3.2 Trajectory generation	53
CHAPTER 6: IMPLEMENTATION OF A EXTERNAL AXES	56
6.1 Approach 1	56
6.1.1 Concept	56
6.1.2 Steps to design the trajectory	58
6.1.3 Examples	59
6.2 Approach 2	71
6.2.1 Concept	71
6.2.2 Steps to design the trajectory	72
6.2.3 Example	73
6.3 Discussion	74
CHAPTER 7: CONCLUSIONS	75
CHAPTER 8: FURTHER DEVELOPMENTS	76
CHAPTER 9: BIBLIOGRAPHY	77
CHAPTER 10: ANNEX	80
10.1 Relative velocity between two particles	80
10.2 Using auxiliary frames	81
10.3 Nearest neighbors algorithm	82
10.4 Programs for planar trajectories	83
10.5 Programs for the external axis	84

0. Motivation

Nowadays, the use of robots for the automation of process is very common. This is due to the advantages provided: cost reduction, quality increase, high reproducibility, etc. Nevertheless the robots have the disadvantage, that a high initial investment is necessary.

Thermal spraying processes use industrial robots for many reasons, some of them are: high control of the process, quality increase, dangerous work environment, etc. The industrial robot can control many parameters during the process; like the trajectory and the velocity of the torch, which have a significant influence on the heat and mass transfer to the piece and coating. Properties such as coating thickness, porosity, micro hardness and thermal stress distribution are therefore significantly influenced by the spraying distance, velocity and trajectory. It is thus necessary to implement new tools, which support robot programming and fulfill the requirements of torch handling for thermal spraying and lacquered operation.

Optimized robot programming is necessary for high quality products regarding coating properties and functionality. To optimize the robot programming, different off-line programming tools are used. The off-line programming has the advantages: increase of work safety and efficiency, low time to program, continuous production, etc.

In some cases, pieces with complex geometries cannot be coated completely using a 6-axis robot. Implementing an external axis, which brings the piece in rotation, allows the complete coating of the piece in one operation.

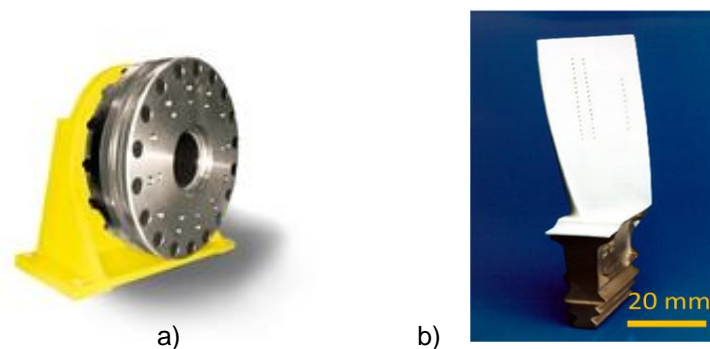


Fig. 0.1: a) External axis from Fanuc; b) Turbine blade.

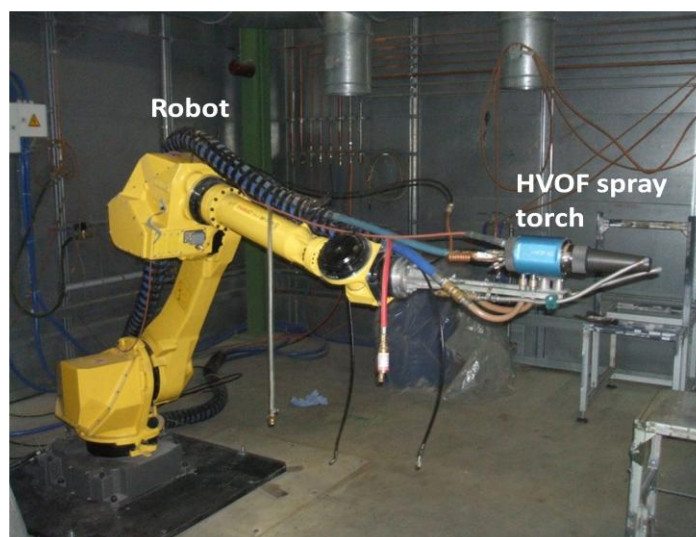


Fig. 0.2: Robot Fanuc M710iC for coating in IFKB.

1. Thermal Spraying Automation

1.1 Principles of Thermal Spraying

The earliest records of thermal spray originate in the patents of M.U. Schoop (Zurich, Switzerland), dating from 1882 to 1889. These patents describe a process that consisted to melt a metal and spraying the atomized metal, through a compressed gas, on a surface, which is going to be coated. In other words, the thermal spraying process is a contribution of atomized spraying materials on the surface, which has been properly prepared [SCHNEIDER06].

A technical definition of thermal spraying can be: “A group of processes in which finely divided metallic or nonmetallic surfacing materials are deposited in a molten or semi-molten condition on a substrate to form a spray deposit. The surfacing material may be in the form of powder, rod, cord, or wire” [WEBASM]. This definition is given by ASM (American Society of Materials).

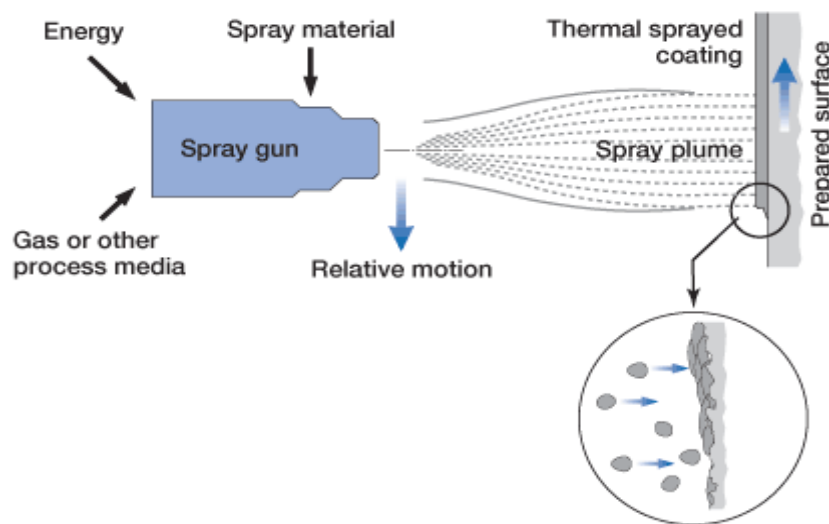


Fig. 1.1: Scheme of the thermal spraying process [WEBSULZER].

Properly applied thermal spray coatings have many uses and offer several advantages [DAVIS2004]:

- **A wide range of materials can be deposited as coatings.**
- **Low Processing Costs:** rapid spray rates and coating deposition result in relatively low processing costs.
- **Wide Range of Coating Thicknesses:** coating thicknesses from 25 μm to 6500 μm (0.002 to 0.250 in.) are used today.
- **Wide Application Range:** thermal spray coatings function effectively for a broad range of applications (Atmospheric and aqueous corrosion control, Metal and ceramic-matrix composite structures, ...)
- **Minimal Thermal Degradation to Substrate:** with proper control, there is little risk of thermally degrading the substrate during spraying.

Thermal spray coatings also have disadvantages, and these are important to understand so that designers can avoid placing thermal spray coatings in situations where satisfactory performance is unlikely. The disadvantages are [DAVIS2004]:

- **Low Bond Strength:** the tensile bond strength achieved via thermal spraying is relatively low compared with other coating processes.
- **Porosity:** coatings are generally somewhat porous, allowing the passage of gases and/or liquids through to the coating/substrate interface.
- **Anisotropic Properties:** thermal spray coatings are anisotropic; that is, they tend to have up to 10 times the tensile strength in the longitudinal direction than in the direction parallel to the spray.
- **Low Loading Capacity:** thermal spray coatings are not generally used as structural members and do not perform adequately under point or line loadings.
- **Line-of-Sight Process:** thermal spray deposition of coatings on simple geometrical configurations is relatively straightforward, but complex shapes or contours, as in the case of gas-turbine airfoils, require expensive, automated, computer-controlled, multi-axis manipulators or robots.

1.2 Thermal Spraying Processes

As saw in previous section, the thermal spraying processes consist in melting a metal and spray this metal on the surface of a material. Depending on the used method to melt the material, the processes are classified. In the European standards the EN657 as well as in the equivalent international standard ISO 14917, the different processes are classified in:

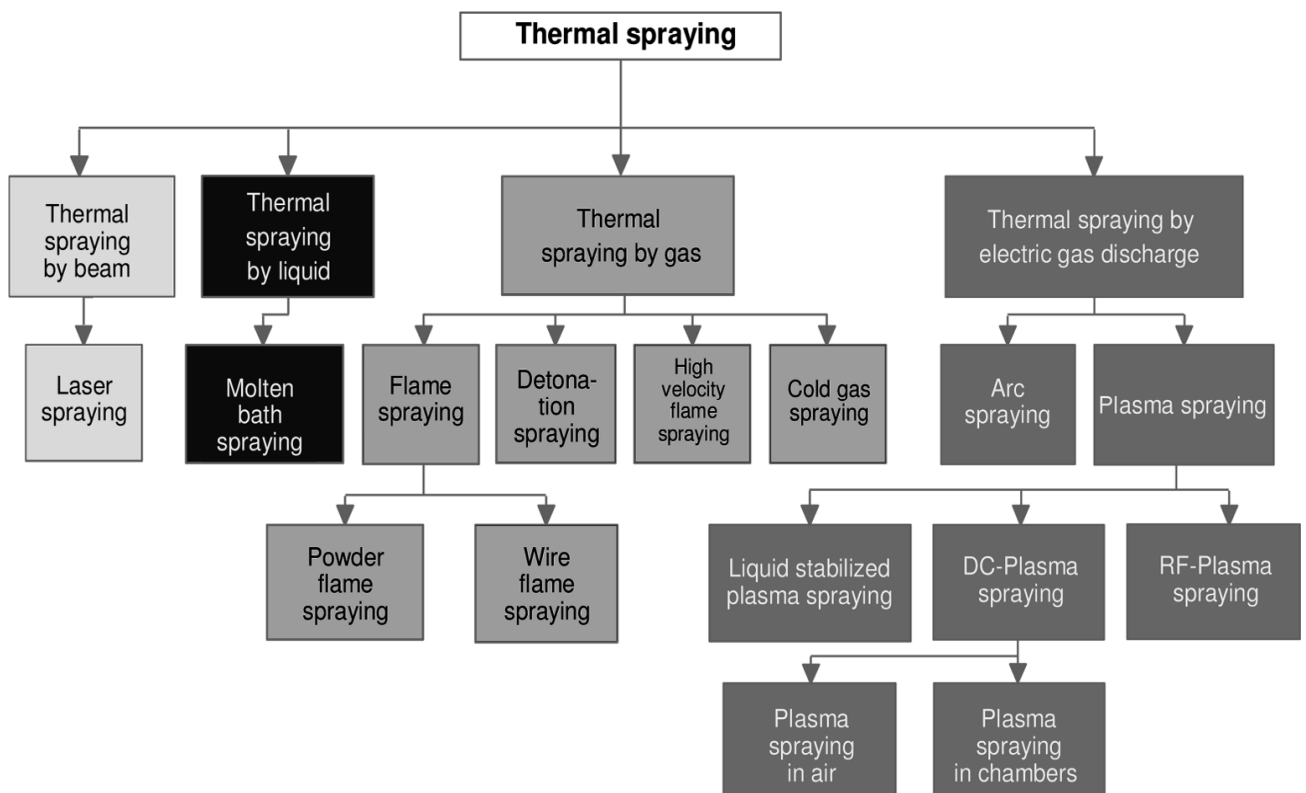


Fig. 1.2: Overview of the different thermal-spray processes in analogy to EN 657 [SCHNEIDER2006].

Below, the main thermal spraying processes are described:

- Thermal Spraying by Beam (Laser Spraying):** Laser coating is an overlay deposition process, where the coating material, a powder or wire, is applied on the surface of the base material through a melting process. In this case the source to melt the material comes from a laser [VUORISTO]. The scheme with the principle of Laser Spraying is showed in figure 1.3.

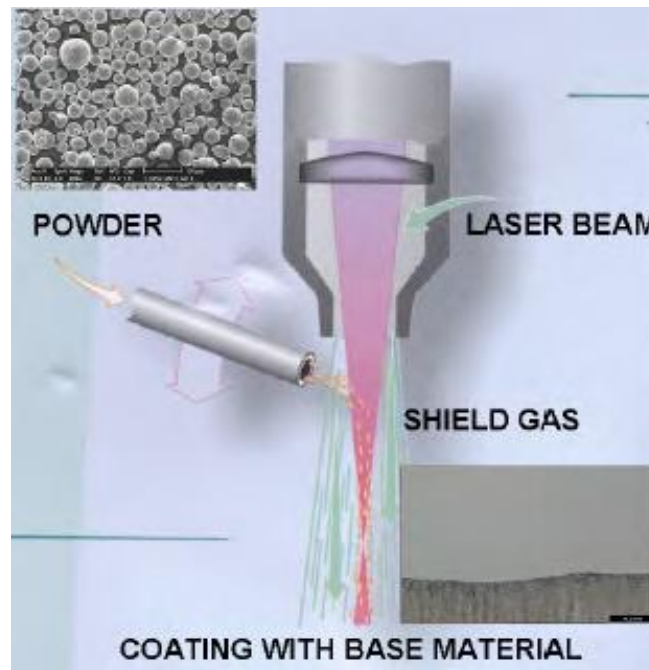


Fig. 1.3: Scheme Laser Spraying [VUORISTO].

- Flame Spraying:** the coating material is melted by a flame of fuel gas and oxygen. Depending on the way the coating material is fed, it can be two processes:
 - PFS (Powder Flame Spraying):** the powder melts in the flame and accelerates to the substrate by fuel gas and additional spray gas.
 - WFS (Wire Flame Spraying):** the coating material is fed in the form of a wire which is melted in the surrounding flame, and then accelerated by the additionally added compressed air towards the substrate.

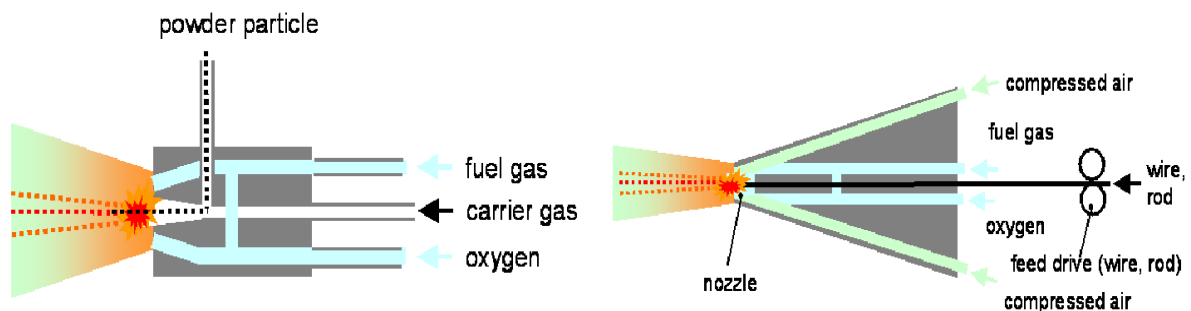


Fig. 1.4: Flame Spraying processes (powder vs. wire) [ACEDO05].

- D-Gun (Detonation Gun):** an explosive mixture of fuel, oxygen and powder is introduced into a long barrel and ignited by a spark plug. A scheme about this process is showed in figure 1.5.

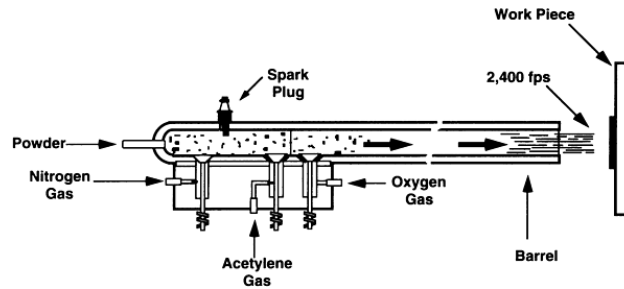


Fig. 1.5: Detonation Gun spraying process [DAVIS04].

- HVOF (High Velocity Oxy-Fuel):** it is similar to the D-Gun spray as it also has an extended confined combustion, but the difference is the fact that HVOF operates on a steady state basis [DAVIS04].

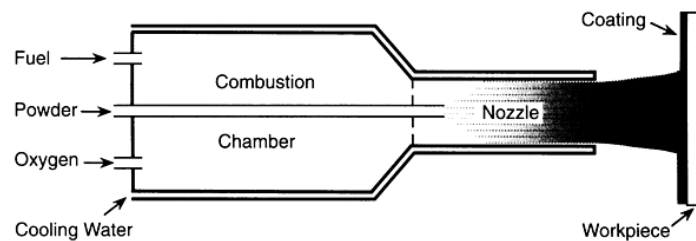


Fig. 1.6: High Velocity Oxy-Fuel process [DAVIS04].

- Cold Gas Spraying:** The powder particles are metered into the gas flow immediately upstream of the converging section of the nozzle and are accelerated by the rapidly expanding gas. The incoming compressed gas can be introduced at room temperature, or it can be preheated in order to achieve higher gas flow velocities in the nozzle [DYKHUIZEN98].

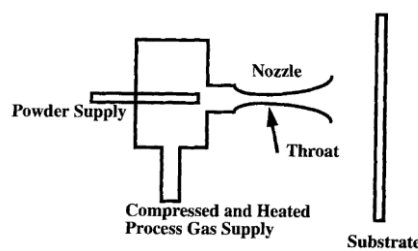


Fig. 1.7: Cold gas spraying process [DYKHUIYEN98].

- **Arc Spraying:** the coating material in the form of two electric wires at two different potentials. A DC electric arc melts and propels the particles toward the substrate.

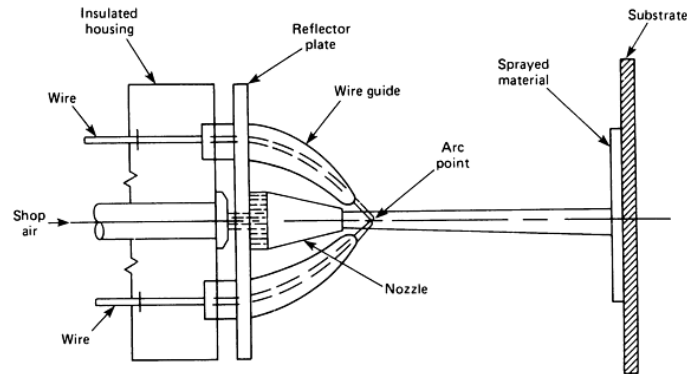


Fig. 1.8: Wire electric arc spraying [DAVIS04].

- **Plasma Spraying:** The plasma state is generated by imparting energy to a gas until this energy is sufficient to ionize the gas and create the plasma. The spray material is then injected into the plasma to be molten and accelerated [DAVIS04]. There are many different types of plasma spraying, the most important are:
 - **APS (Atmospheric Plasma Spraying)**
 - **ICPS (Inert Chamber Plasma Spraying)**
 - **RF (Radiofrequency Plasma)**
 - **SPS (Shrouded Plasma Spray)**

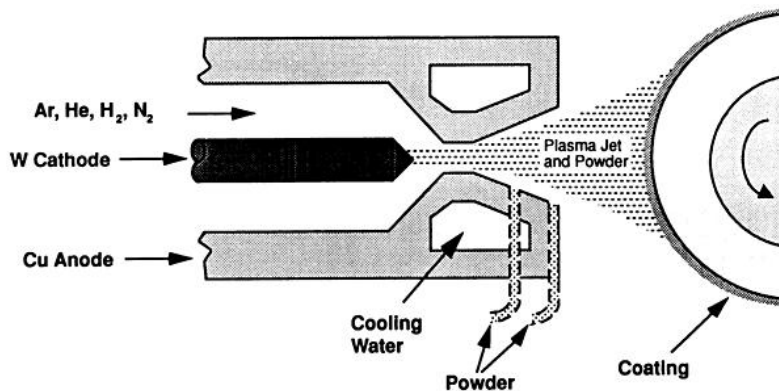
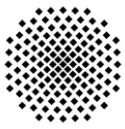


Fig. 1.9: Plasma spraying process [DAVIS04].



In next table the specific process attributes and the resulting coating characteristics are compared of the thermal spraying processes.

Process	Materials	Feed Material	Surface Preparation	Substrate Temperature (°C)	Particle Velocity (mm/s)
Powder Flame Spray	Metallic, ceramic and fusible coatings	Powder	Grit blasting or rough threading	105 - 160	65 - 130
Wire Flame Spray	Metallic coatings	Wire	Grit blasting or rough threading	95 - 135	230 - 295
Ceramic Rod Spray	Ceramic and cermet coatings	Rod	Grit blasting	95 - 135	260 - 360
Two-wire electric-arc	Metallic coatings	Wire	Grit blasting or rough threading	50 - 120	240
Nontransferred arc plasma	Metallic, ceramic, plastics and compounds	Powder	Grit blasting or rough threading	95 - 120	240 - 560
High-Velocity Oxyfuel	Metallic, cermet and some ceramic	Powder	Grit blasting	95 - 150	100 - 550
Detonation Gun	Metallic, cermet and ceramic	Powder	Grit blasting or as-machined	95 - 150	730 - 790
Super D-Gun	Metallic, cermet and ceramic	Powder	Grit blasting or as-machined	95 - 150	850 - 1000
Transferred arc plasma	Metallic fusible coatings	Powder	Light grit blasting or chemical cleaning	Fuses base Metal	490

Table 1.1: Thermal spray processes comparison [DAVIS04].

1.3 Use of Robots in Thermal Spraying

Up to now, the principles about thermal spraying and the different processes in thermal spraying have been seen. In this section, the reasons, which justify the use of robots in thermal spraying processes, are showed.

In thermal spraying processes, the pieces, which are coated, have to possess requirements for future use. In other words, these pieces have to achieve quality requirements. To obtain this quality, in the thermal spraying process, is necessary that the coated process would be accurate.

Thermal spraying processes are industrial processes and the coating is not going to be realized in only one piece; but the coating process will be for a batch of pieces. This causes the process has to be repetitivity.

By the moment, the thermal spraying processes have to be accurate, to achieve the quality requirements and repetitivity. If to the previous conditions, the fact that the work environment is harmful to humans (gases, high temperatures, micro-particles, etc.) is added, the use of robots is justified for thermal spraying processes.

In resume, industrial robots are used in thermal spraying processes, because of the next reasons:

- Accuracy.
- Quality.
- Repetitivity.
- Worker protection (Harmful work environment).

As said before, the quality of the piece is obtained with accurate control of the robot and spray parameters. In the process there are many parameters, which influence in the quality of the piece. Just a few parameters can be controlled by the robot. These parameters are:

- **Spray Velocity:** the linear velocity at which the thermal spraying gun focus covers along the surface of the work piece.
- **Spray Distance:** distance from the extreme of the gun until the coating surface.
- **Spray Angle:** angle between the axis of the gun and the tangent line to the substrate surface in the coating point.
- **Trajectory:** number of ordered points to follow to coat the piece. It can have different shapes. The most common path for this kind of process is the meander (see chapter 4).

All this parameters can be seen in the next figure:

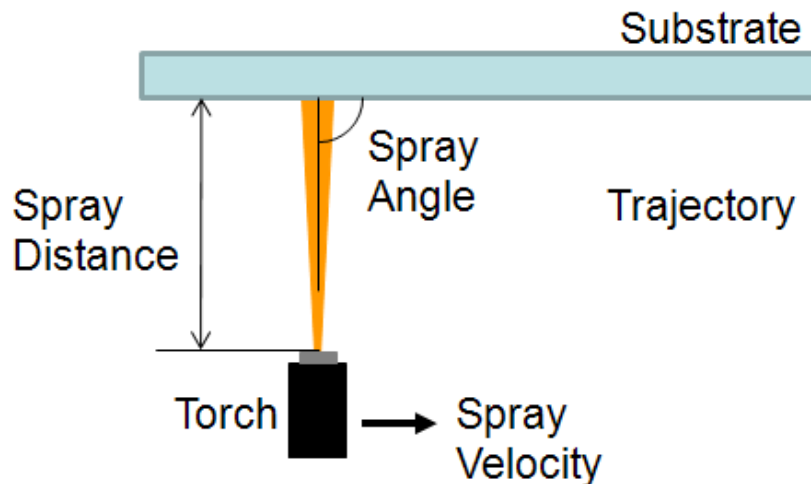


Fig. 1.10: Parameters in thermal spraying processes controlled by the robot.

These parameters influence the mass transfer (amount of material deposited on the substrate) and the temperature on the surface during the process. The mass transfer and the temperature on the material are features, which influence the quality of the obtained layer. Therefore, a precise control of the previous parameters will allow a higher quality of the product.

The quality on the surface has to be homogeneous. To obtain this, the parameters spray velocity, spray distance and spray angle; have to be constant along the trajectory. Keeping these parameters constants along the trajectory is not easy task if the piece, which has to be coated, has a complex geometry (figure 1.11).



Fig. 1.11: Sigma-form blade for injection molding application.

Several studies have been developed in this area; to achieve complicated pieces coating (with complex geometry) and keep constant the parameters (coating velocity, coating angle and coating distance). Some of these studies have been helpful to realize this project [FRUTOS09] [CANDELDIP].

Finally, the kind of robot used normally, is an articulated robot. The use of this robot is due to its high degree of freedom (6 degrees of freedom); which an easier adaptation to pieces with complex geometry. In the figure 1.12, the robot RX130, which is used in coating operations in IFKB, is showed.



Fig. 1.5: Robot RX 130 (Co. Stäubli) for the coating in IFKB.

2. Industrial Robots for surface technologies

Nowadays the use of robots for many tasks is usual. Robots are generally used to perform unsafe, hazardous and highly repetitive tasks. There is a big range of fields of application of robots, some of them are: medical laboratories, medicine, nuclear energy, agriculture, spatial exploration, underwater inspection, arts and entertainment, automotive industry, etc.

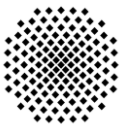
In this work, the applications of the robots in the industry will be considered. Some of the industrial applications are: welding applications, spray painting applications, assembly operations, palletizing and material handling, dispensing operations, water jet cutting, etc.

The aim of the present work is to develop methods of robot programming, based on the geometrical description of the component to be coated or painted, for the optimization of surface coating and painting processes.

In this chapter, a general view of the industrial robots and their main features will be seen.



Fig. 2.1: A KUKA robot drills and installs fasteners in aircraft components [WEBKUKACOMP].



2.1. Definition of Industrial Robot

There are many definitions of robots; and these definitions depend mainly on the task of the robot. Between all the definitions the definitions of robots related with Industrial applications have been selected. Some of these definitions for an Industrial Robot are:

“An automatically controlled, **reprogrammable, multipurpose manipulator** programmable in three or more axes, which may be either, fixed in place or mobile for use in **industrial automation applications** [ISA8373].”

“A robot is a **reprogrammable, multifunctional** machine designed to **manipulate** materials, parts, tools, or specialized devices, through variable programmed motions for the performance of a **variety of tasks** [RIA].”

According to these definitions, it can be said that industrial robots are normally formed by one multi-jointed arm with a wrist at the end and controllers that have a program memory and can be able to accept signals from external sensors. An industrial robot will be used in highly repetitive tasks.

2.2. Classification of Robots

In this section, the classification of robots is going to be focused on industrial robots. The industrial robots can be classified according to various criteria, some of them are:

- Degrees of freedom.
- Drive technology.
- Level of autonomy.
- Workspace geometry.
- Etc.

The classification according to degrees of freedom makes reference to the quantity of degrees of freedom, which have the robot. There are robots with a great variety of degrees of freedom, from the elemental robot (3 or 4 degrees of freedom) until the articulated robot with 6 or even more degrees of freedom.

Another classification is according to the drive technology. The drive technology is the power source used for the movement of the robot. The power source of the robot can be: electrical, pneumatic, hydraulic and combustion engine.

The classification according to the level of autonomy of the robot takes in account the interaction between human control and the machine motion [BARRIENTOS07]. The different levels of autonomy are:

- **Teleoperation:** a human controls each movement, each machine actuator change is specified by the operator.
- **Supervisory:** a human specifies general moves or position changes and the machine decides specific movements of its actuators
- **Task-level autonomy:** the operator specifies only the task and the robot manages itself to complete it.

- **Fully autonomy:** the machine will create and complete all its tasks without human interaction.

Finally one of the most important classifications of the robots is according to the workspace geometry. The robot workspace (sometimes known as reachable space) is all places that the end effectors (gripper) can reach. The workspace is dependent on the DOF (degrees of freedom), angle/translation limitations, the arm link lengths, the angle at which something must be picked up at, etc. Generally, there are five kinds of robots depending of their workspace geometry:

- **Cartesian Robot:** robots whose arms have three prismatic joints, whose axes are coincident with a Cartesian coordinate system [ISA8373].
- **Cylindrical Robot:** robots whose arms have two prismatic joints and one revolute joint.
- **Spherical Robot:** robots whose arms have one prismatic joint and two revolute joints but not paralleled.
- **SCARA Robot:** (Selectively Compliant Arm for Robotic Assembly) robot which has two rotary joints to provide compliance in a selected plane [ISA8373].
- **Articulated Robot:** robots whose arms (primary axes) have three concurrent rotary joints [ISA8373].

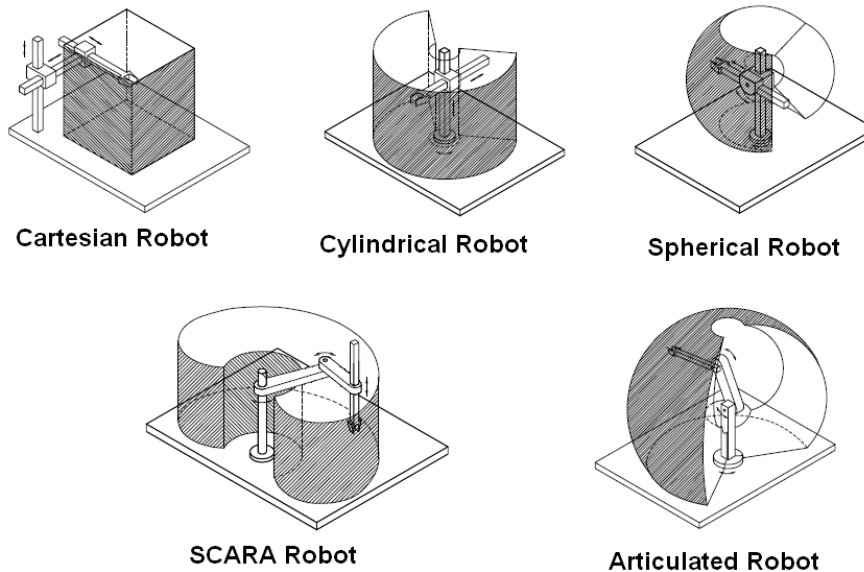


Fig. 2.2: Classification of robots according with the workspace [SICILIANO09].

2.3. Basic Components of Industrial Robots

The industrial robots can be divided in many different parts, depending on many factors. In this project, a deep study about the industrial robots is not the objective. The main subsystems of an industrial robot are [KERAMAS99]:

- Manipulator (Mechanical Unit)
- Power Supply
- The end – effector (Robot Tooling)
- Robot Controller

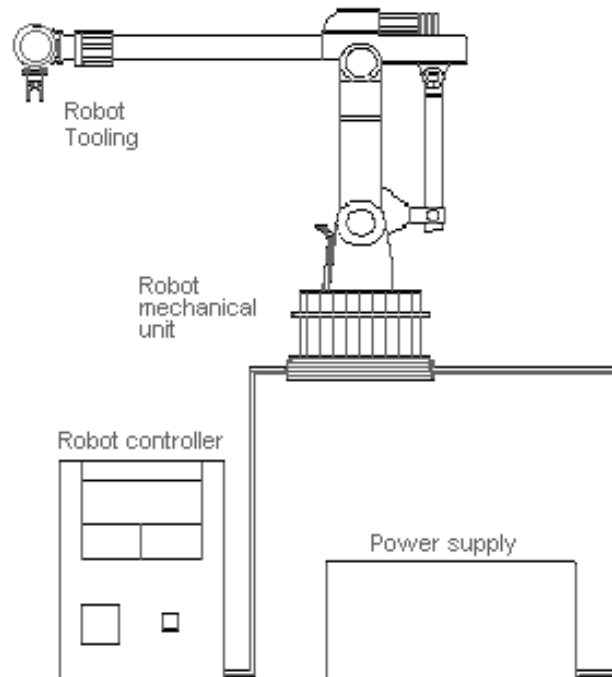


Fig. 2.3: Basic Components of Industrial Robots [KERAMAS99].

2.3.1 Manipulator or Mechanical Unit

The manipulator, which is the robot's arm, consists of segments jointed together with axes capable of motion in various directions allowing the robot to perform work. Its primary function is to provide the specific motions that will enable the tooling at the end of the arm to do the required work [CRAIG90].

This mechanical unit is also comprised of a fabricated structural frame with provisions for supporting mechanical linkage and joints, guides, actuators (linear or rotary), control valves, and sensors. The physical dimensions, design, and weight-carrying ability depend on application requirements.

The manipulator can be divided in three parts: body, arm and wrist (figure 2.4). At the same time the movement of a manipulator can be divided in two: arm and body motions, and wrist motions. The arm and body motions will position the extreme of the manipulator in the space; meanwhile the wrist motions will orient the extreme of the manipulator [KERAMAS99].

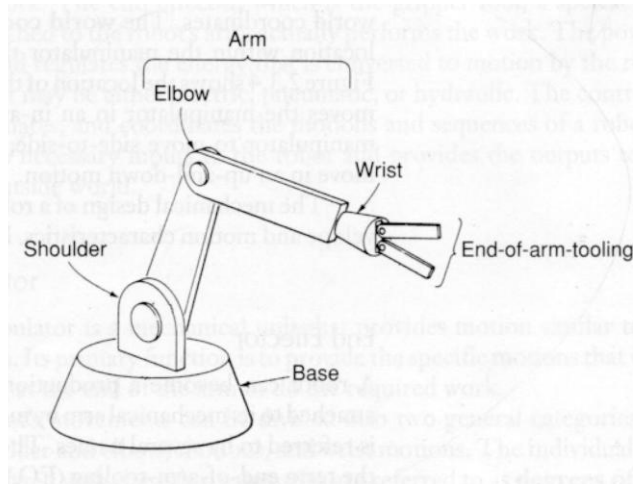


Fig. 2.4: Parts of a manipulator [KERAMAS99].

2.3.2 The Power Supply

The function of the power supply is to provide and regulate energy that is required for a robot to be operated. There are three basic types of power supplies [BARRIENTOS07]:

- Electric
- Hydraulic
- Pneumatic

Some robot systems can require a combination of the three sources. The majority of the industrial robots are powered by electricity; meanwhile hydraulic and pneumatic powers are relatively rare. The electric power can be subdivided into:

- DC electricity: make possible a smooth and continuously controllable movement.
- AC electricity (synchronous and asynchronous): is used normally in applications with high torque and rotating rate, where adequate dynamic behavior is required.

2.3.3 The End – Effectors

The end-effector is a device mounted on the tool plate of the robot's wrist to perform a particular task. Depending on the type of operation, the end-effectors can be:

- Grippers, hooks, scoops, electromagnets, vacuum cups and adhesive fingers for material handling.
- Spray gun for painting.
- Attachments for spot and arc welding and arc cutting.
- Power tools such as drills, nut drivers and burrs.
- Special devices and fixtures for machining and assembly.
- Measuring instruments, such as dial indicators, depth gauges, etc.

In some cases, the end-effector can have its own degrees of freedom, and then the total number of degrees of freedom is increased. So, a robot manipulator with 6 degrees of freedom that is fitted with an end-effector, which itself has 2 degrees of freedom, forms a 8 degrees of freedom system [SICILIANO09].

A good end-effector or terminal device should be:

- Very reliable.
- Low weight.
- As small and compact as possible.
- Easy to change and maintain.

2.3.4 Robot Controller

The controller is a communication and information processing device that initiates, terminates and coordinates the motions and sequences of a robot. It accepts necessary inputs to the robot and provides the output drive signals to a controlling motor or actuator to correspond with the robot movements and outside world [KERAMAS99].

The robot controller generally includes:

- A central processing unit (CPU), which is the brain of the robot.
- Memory to store the control program and the state of the robot system obtained from the sensors.
- The appropriate hardware to interface with the external world (sensors and actuators).
- The hardware for a user interface.

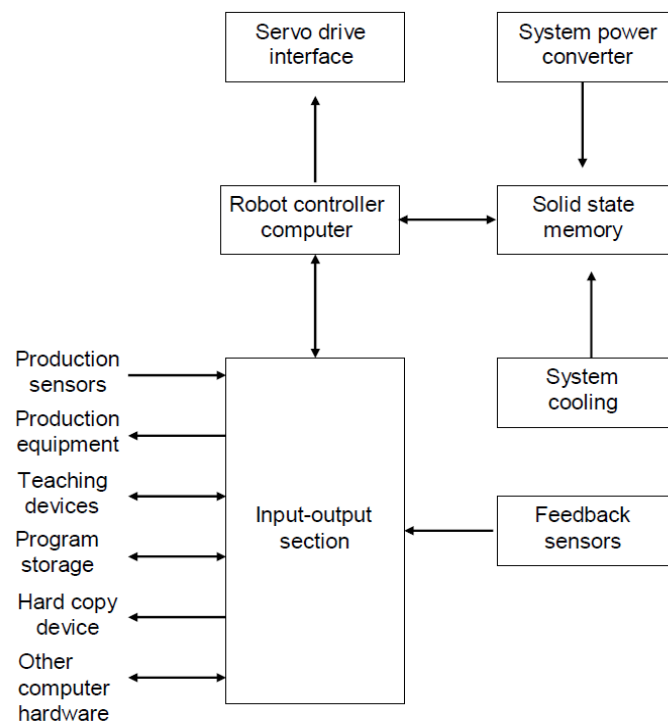


Fig. 2.6: Robot controller block diagram [KERAMAS99].

2.4. Coordinate Systems

A coordinate system defines the position and attitude of the robot. The reference systems are usually three:

- **World-System:** is predefined for each robot and it is used as the default frame of reference (figure 2.7).
- **Tool-System:** has its origin at the center of the faceplate surface of the robot. Its orientation is defined with the plane of the x-axis and y-axis on the faceplate and the positive z-axis pointing straight out from the faceplate (figure 2.7).
- **Joint-System:** is defined for robot joints. The position and attitude of the robot are defined by angular displacements with regard to the joint coordinate system of the joint base (figure 2.8).

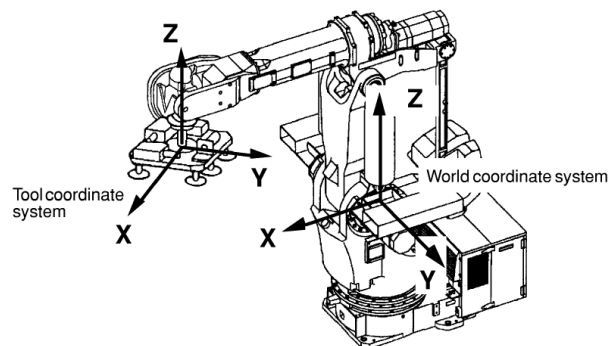


Fig. 2.7: World and Tool Coordinate Systems [MANHANDLING].

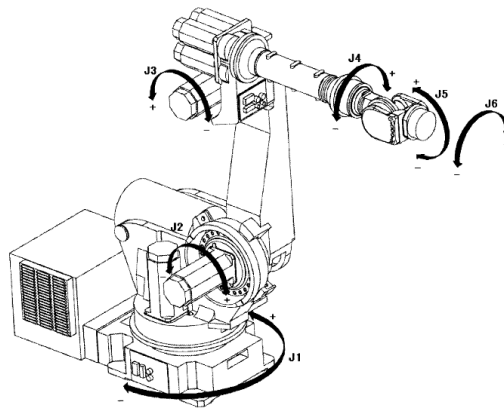


Fig. 2.8: Joint Coordinate System [MANHANDLING].

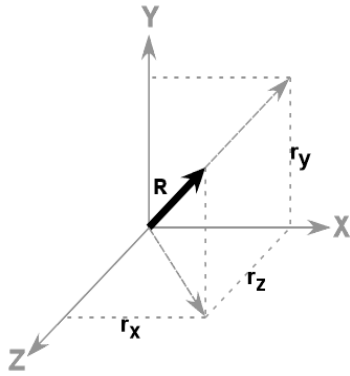
Once a reference coordinate system has been established, the next step is to identify the location of a body in the space. The location of a body is divided in two parts: position and orientation. The three most common coordinate systems used to define position are [BARRIENTOS07]:

- **Cartesian coordinates:** the position of a point, with respect to the coordinate system OXYZ, is defined by three values x , y and z .
- **Cylindrical coordinates:** the position of a point, with respect to the global reference system is defined by the values: magnitude of the projection of vector P onto the

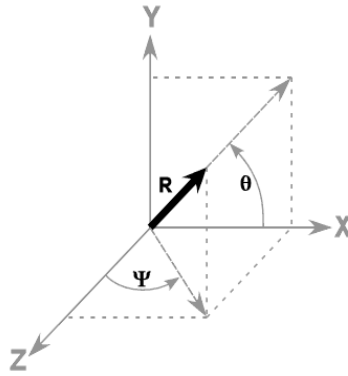
plane OXY, angle between the axis OX and the mentioned projection of P and the magnitude of the projection of vector P onto axis OZ (figure 2.9).

- **Spherical coordinates:** in this case, the values are the magnitude of the projection of vector P onto OXY, the angle between the axis OX and the projection of vector P onto plane OXY, the angle between axis OZ and vector P (figure 2.9).

Cartesian Components (x,y,z)



Spherical Components (r, q, Y)



Cylindrical Components (r, Y)

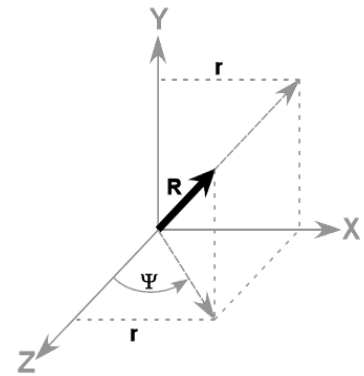


Fig. 2.9: Coordinates systems for position.

On the other hand, the most common systems used to define orientation are:

- **Rotation Matrix:** defines the orientation of one system over another, and it is used to transform the coordinates of a vector in one system to another. Also it is called the matrix of direction cosines.
- **Euler's Angles:** minimal representation of relative orientation. This set of three angles set describes a sequence of rotations about the axes of a moving reference frame. There are, however, many (12, to be exact) sets that describe the same orientation: different combinations of axes (ZXZ, ZYZ, etc.) lead to different Euler angles.
- **RPY Angles:** three consecutive rotations about fixed axes instead of moving axes. This leads to a notation with **Roll-Pitch-Yaw** (RPY) angles.

In industrial robot normally, the location can be considered as composed of the position of a point (x, y, z) in a Cartesian system and the orientation of the moving frame with respect to the fixed frame (w, p, r), the last three components specify the orientation of the end-tooling (figure 2.9). The three angles are defined as:

- **Yaw:** is a rotation about the local reference frame Z axis.
- **Pitch:** is defined as a rotation about the local reference frame Y axis, after yaw has been applied.
- **Roll:** is defined as a rotation about Z axis of the local reference frame after yaw and pitch have been applied.

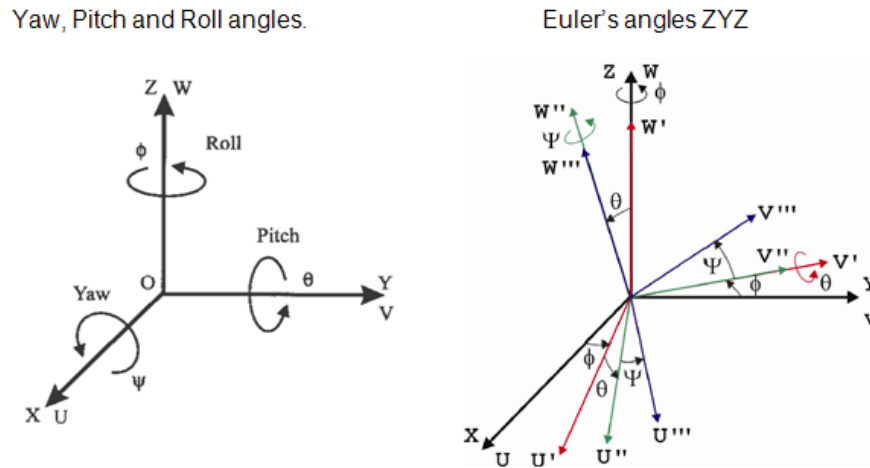


Fig. 2.9: Systems of Orientation [BARRIENTOS07].

2.5 Robots Kinematics

Kinematics is the science of motion that treats the subject without regard to the forces that cause it. Within the science of kinematics, one studies the position, the velocity, the acceleration, and all higher order derivatives of the position variables (with respect to time or any other variable(s)). Hence, the study of the kinematics of manipulators refers to all the geometrical and time-based properties of the motion. Meanwhile, the dynamic is the study of movement or motion with regard to the forces or torques that produce it [CRAIG90].

Kinematics of robot manipulator arms can be studied without needing to study their dynamics, because all the forces involved are generated and controlled at the basic level of control, and above this level, these forces and their consequences can be safely ignored, as long as the speed and accelerations of the movements are not high. This is why we can talk of a kinematic level of control, and not a dynamic level of control. It is a situation special to industrial robot

2.5.1 Forward Kinematics

The problem of forward kinematics consists in determining what the position and orientation of the reference system attached at the end-point of the robot is, with respect to some fixed global reference system, given the values of the positions of each the joints, plus information about the types of joints and the geometry of the elements that connect the joints [MCKERRON91].

Given a reference system attached to each link of the robot arm, the position and orientation of the end-point (defined as the position of the origin of its reference system and its orientation with respect to the frame fixed to the base of the robot) can be calculated by the composition of the homogeneous transformation matrices which relate reference system on successive links.

In other words, the problem of forward kinematics can be reduced to the calculation of the relationship between two connected links in a robot arm, so a coordinate system will be fixed to each link, and then construct the homogeneous transformation matrix that represents the geometric relationship between them.

To do this in a convenient and accurate manner, all these link reference systems should be defined in the same way. The Denavit and Hartenberg method is the most used criterion to establish the link reference systems in robot arms [DENAVID64].



2.5.2 Inverse Kinematics

The direct kinematics equation establishes the functional relationship between the joint variables and the end-effector position and orientation. The inverse kinematics consists of the determination of the joint variables corresponding to a given end-effector position and orientation. The solution to this problem is of fundamental importance in order to transform the motion specifications, assigned to the end-effector in the operational space, into the corresponding joint space motions that allow execution of the desired motion [CRAIG90].

To solve the inverse kinematics, two methods can be applied: solution of inverse kinematics by resolving the equation system given by transformation matrix and the solution based on geometrical relationships. The existence of solutions is guaranteed only if the given end-effector position and orientation belong to the manipulator dexterous workspace. Also it could be multiple solutions or even infinite solutions, just in the case of a kinematically redundant manipulator [MCKERRON91].



3. Robot Programming

Until now, the different kinds of robots, its components and the basic concepts have been seen. This chapter focuses on the methods used to program a robot and on the different levels of programming.

Programming is the identification of a series of basic actions which, when executed in the specific order, achieve some specific task or realize some specific process [MCKERRON91].

3.1 Levels of programming

Depending on the different types of basic actions specified in the programs developed, three levels of programming can be distinguished [CRAIG90].

JOINT LEVEL PROGRAMMING

The basic actions are defined in terms of positions of individual joints of the robot arm. In some cases, at this level, basic actions can also be defined in terms of speed of movement of individual joints.

ROBOT LEVEL PROGRAMMING

In this case, the basic actions are defined in terms of position and orientation. At this level, particular kinds of movement can be defined: straight lines, circular arc movements, etc. As in the joint level programming, the speed can be defined. To define these parameters, different languages of programming are used:

- AL, University of Stanford, USA (1974).
- V+, Adept (1989).
- KAREL, Fanuc (1988).
- RAPID, ABB (1994).

HIGH LEVEL PROGRAMMING

The high level programming can be divided in:

- Object-Level Programming: the actions are operations to be performed on the objects to be manipulated or assembled. Some of programming languages are: LAMA, AUTOPASS, RAPT, etc.
- Task-Level Programming: the actions specified in the code are complete task or subtasks.

3.2 Methods of Programming

In order for a robotic manipulator to perform useful work, it must be programmed to accomplish the desired task or motion cycle. Despite the great evolution of the industrial robot, in the majority of the industrial applications, the robot programming is made using one of the following ways [MCKERRON91]:

- On-line programming.
- Off-line programming.
- Hybrid programming.

ON-LINE PROGRAMMING

Programming by guiding or programming by teaching. This involves physically leading (moving) the robot arm through the movements and actions to be performed. Positions (and in some cases trajectories) are recorded by the robot system. Some methods for on-line programming are showed in figure 3.1.

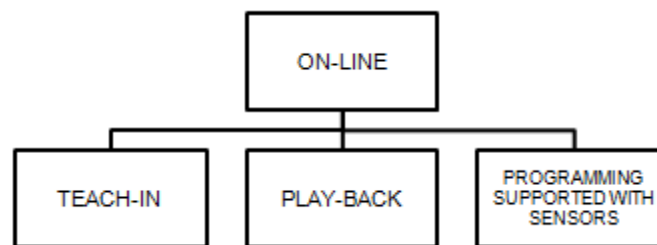


Fig. 3.1: Methods of on-line programming.

TEACH-IN: consists of moving the robot through the task, principally using the manual control pendant and when a desired point is reached, the operator stores coordinates into the robot's memory.

PLAY-BACK: consists on taking the robot directly along the work piece path by the programmer manually.

PROGRAMMING SUPPORTED WITH SENSORS: the robot path is calculated during the movement with the help of the sensors independent of the controller.

Finally in the table 3.1, the advantages and disadvantages are showed:

ADVANTAGES	DISADVANTAGES
Easy to perform.	During programming the robot cannot be used for other tasks.
Accuracy.	The taught sequence cannot be altered.
	Difficult adaption of the programs.
	It cannot be used in hazardous situations.

Table 3.1: Advantages and disadvantages of on-line programming.

OFF-LINE PROGRAMMING

This method involves the development of a text-file, which contains the robot instructions and declarations that form the execution sequence. Indeed, in this text file other information, such as types of trajectories and maximum speed of movement are specified. Robot off-line programming is a method, which combines computer simulation and graphics to produce a desired trajectory plan [MITS104]. The different methods of off-line programming are showed in figure 3.2:

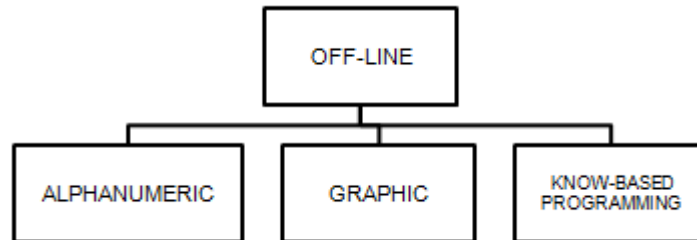


Fig. 3.2: Methods of off-line programming.

ALPHANUMERIC: this programming is based on introducing the robot program by the keyboard in a text editor. It is very important to know the programming language of the robot.

GRAPHIC: it can simulate the working cell of the robot in 3D. With help of a graphic model the robot positions can be visualized in a screen.

KNOW-BASED PROGRAMMING: it is the higher step in off-line programming for industrial robots, which is based on the expert system and the knowledge of work. The expert system studies the task and generates a program for the robot depending on the specific task and the accumulated previous knowledge.

Finally in the table 3.1, the advantages and disadvantages are showed:

ADVANTAGES	DISADVANTAGES
Risk reduction.	High initial expenses.
Efficiency and productivity.	Low precision, inaccuracy of position.
Continuous production.	Calibration.
Software reuse.	Worker formation.

Table 3.2: Advantages and disadvantages of off-line programming.

HYBRID PROGRAMMING

This method utilizes the advantages of on-line and off-line programming to optimize the programming technique [CANDEL05]. A robot program consists mainly of two parts:

- Locations: position and alignment (on-line programming).
- Programs logics: controller, structures, communications and calculations (off-line programming).

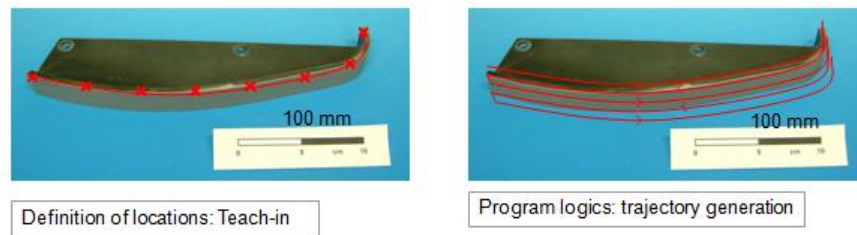


Fig. 3.3: Scheme with the steps of hybrid programming (blade for printing industry).

The program logics can be effectively developed as debugging and simulation facilities are available here. The major part of movement commands can be created off-line by the reuse of CAD data and by interaction of the programmer.

Movement commands to locate the placement of the piece in the robot's workcell are programmed on-line, to achieve a high accuracy in the trajectory. In this manner both methods advantages can be utilized.

The usage of hybrid programming is a very practical way of increasing flexibility in production and thereby increasing the effect of robot manufacturing. In the manner rearrangement time can substantially be reduced, allowing for cost effectiveness in production of even small batches.



4. Design of Trajectories for planar substrates

In the previous chapters, the basic robot theory has been explained. Chapter 4 is going to deal with the techniques which have been used in this project to design trajectories for industrial robots. For each technique, the steps followed to create a program for the trajectory are going to be explained in detail. At the end, a program will be obtained with the desired trajectory.

The trajectories, which are going to be designed, will focus on thermal and lacquer spraying processes. The applications of each trajectory will be seen in the section 4.4 in detail.

Different kinds of programming have been seen in chapter 3. The three different ways to program are: on-line, off-line and hybrid. It makes only sense to speak about design of trajectories for off-line and hybrid programming. This is because the online programming is based on, as seen in chapter 3, teaching a number of points manually to define the trajectory, without needing a program to calculate this points. In consequence, just the off-line and hybrid programming will be considered for the development of robot programs for thermal and lacquer spraying operations.

4.1 Steps to Create a Program

In this section the different steps from the specifications of the trajectory until the robot program, are going to be explained in detail. Different kinds of software have been used to develop the programs:

- Matlab.
- Roboguide.
- Adept.

Matlab is used to calculate and check the trajectory, through graphs. Meanwhile, Roboguide and Adept are used to load and check the trajectory in the robot. The steps to design a program without using a simulation software are showed in figure 4.1, while the steps to design a program with using a simulation software are showed in figure 4.2.

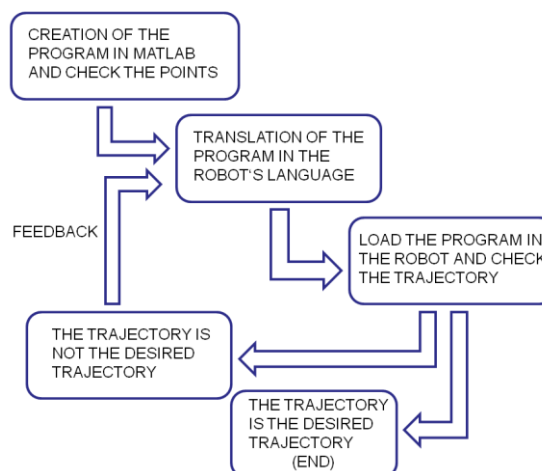


Fig. 4.1: Steps to create a program without simulation software.

The steps “creation of the program in Matlab”, “translation of the program in the robot’s language” and “load the program in the robot”, are common to both schemes. The difference is going to be the platform, where the program can be checked: in the real robot (no simulation software) or in the simulated area (simulation software).

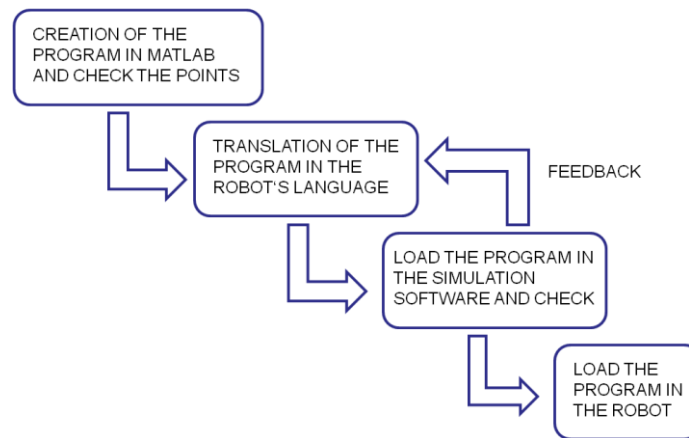


Fig. 4.2: Steps to create a program with simulation software.

The first step is the creation of the program in Matlab. This step is done to create a first version of the program; which could be modified, visualized and changed easily. With Matlab the graphs and the points of the trajectory can be obtained with mathematical equations and logical structures. The structure, which is going to follow the matlab program for designed trajectories, is showed in figure 4.3.

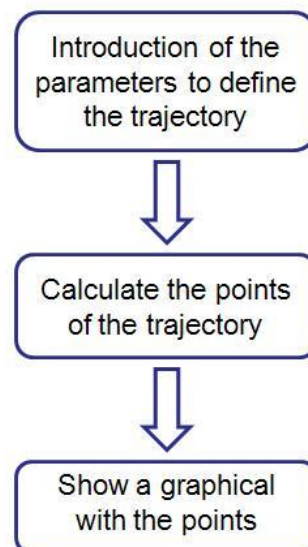


Fig. 4.3: Structure of the matlab program.

In the first part of the matlab program, the parameters, which define the trajectory, are introduced. These parameters could be: first point, offset meander, number of cycles, etc. Some of these parameters are specific for thermal and lacquer spraying processes. Once the parameters are introduced, the next step is to calculate the points of the trajectory. In this part the mathematical equations and logical structures are used to get the points. Finally, to make sure that the calculation of the points is right, the trajectory of the robot is showed in a graph. If the trajectory, which is showed in the graph, is correct; it will be possible to follow the next step: translation of the program in the robot language.



The translation to robot language changes the grammatical structure of the program. Unfortunately, there is not a universal robot language, so depending on the brand of the robot, the language will be different. In the section 4.3, the different languages and the robots, which are going to be used, will be explained in detail.

On the other hand, for translating to robot language, it is also necessary to change the system of reference. In the matlab program, the points of the trajectory are calculated based on the matlab system of reference. For changing from the system of reference defined in matlab to the system of reference of the robot, a series of mathematical calculations have to be done. The basic knowledge needed to translate from one system of reference to other, is explained in the appendix 10.2.

Finally, once the translation to the robot language has been done, and the program has been checked. The program is ready to use in the robot and the lacquer or thermal spraying operations can be done with a real piece.

4.2 Description of applications

In this section, the different application of the trajectories will be explained. The objective of one application is to test different trajectories on a piece and check the coating microstructure and functionality of the substrate:

- Temperature distribution.
- Porosity.
- Thickness.
- Residual Stresses.

The trajectories, which have been programmed, are:

- Meander A.
- Meander B.
- Meander C.
- Triangular trajectory (Mirror)

Another application will be to control the layer thickness of the substrate. The definitive results will be a constant layer and a gradient layer thickness (figure 4.24). To obtain these results, two different kinds of trajectories have been designed:

- Meander A = Constant layer.
- Variable meander = Gradient layer.

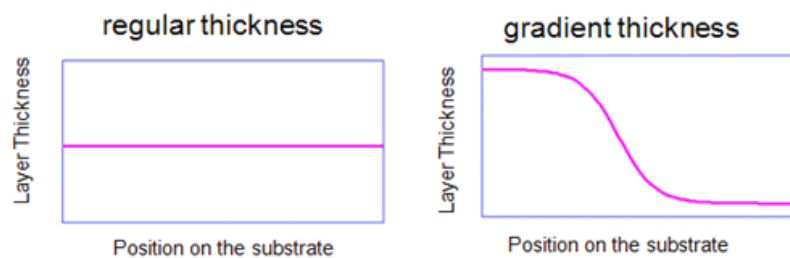


Fig. 4.24: Scheme of the desired results in the layer thickness.

This section is divided in five parts; each part is going to describe one kind of trajectory. The results obtained in this section are the graphs with the points of the trajectory.

4.2.1 Meander A

The first trajectory typical for thermal spraying operations is a meander movement which is showed in Fig. 4.4.

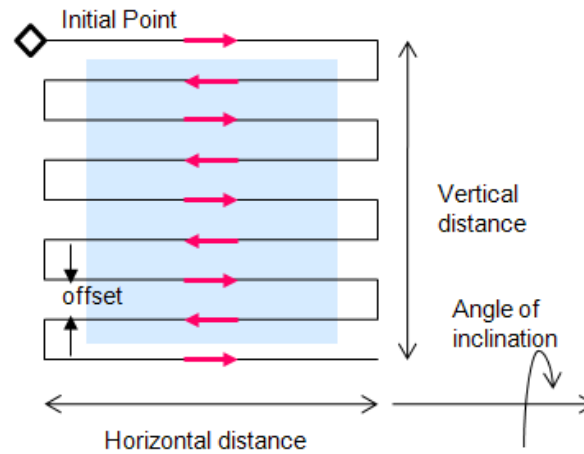


Fig. 4.4: Trajectory Meander A.

The parameters, which are going to define this trajectory, are:

- Initial point: first point in the upper left corner.
- Number of vertical divisions.
- Vertical distance.
- Horizontal distance.
- Angle of inclination: angle of rotation around the axis defined by the horizontal distance.
- Increase distance: this parameter permits modifying the spray distance without modifying the first point of the trajectory.

The parameter offset has not been included; because it can be obtained with the vertical distance and the number of vertical divisions. The equation which relates these three parameters is:

$$offset = \frac{vertical\ distance}{number\ of\ vertical\ divisions}$$

The angle of inclination is a parameter added for trajectories in thermal spraying processes. As seen in section 1.3, one of the parameters important in the thermal spraying processes is the angle of the gun respect to the piece (spray angle). To keep this angle constant independently of the inclination of the piece; the parameter angle of inclination has been added. In consequence, changing this parameter is possible to change the points of the trajectory for obtaining a trajectory parallel to the piece surface.

In the following figure, four meanders are showed, each one with a different angle of inclination (0°, 30°, 60° and 90°).

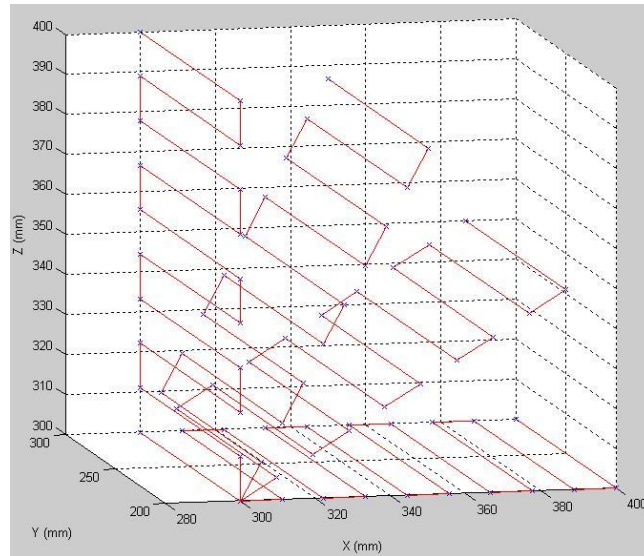


Fig. 4.5: Meanders A with different angles of inclination.

The parameter increase distance has been introduced to make it easier to change the distance of coating, which is a very important parameter in thermal and lacquer spraying processes. In consequence, the trajectory is shifted along an axis perpendicular to the surface created by the trajectory. In figure 4.6, two meanders with the same parameters except for the increase distance, which is 0 and 100mm in each one, are shown.

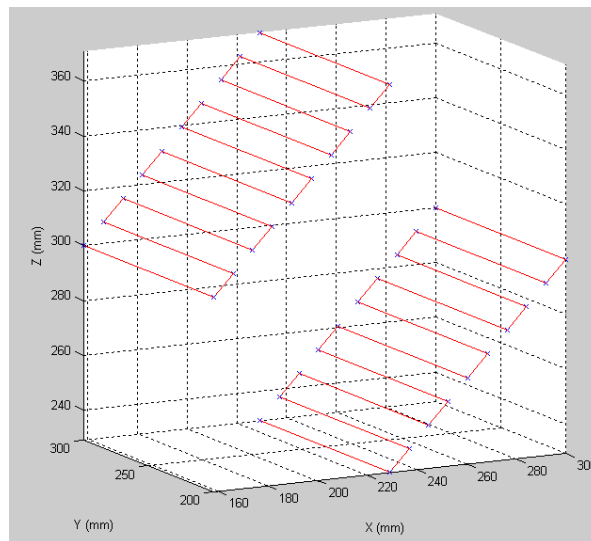


Fig. 4.6: Meanders A with different increase distances.

Finally, the trajectory of the robot is visualized graphically, and it is possible to check in an easy way if there are problems with the points. Once the program has been checked, it is possible to pass to the next step, which consists in translating the program to the robot language.

4.2.2 Meander B

The meander B is similar to the meander A, but the difference is the order of the points to cover of the trajectory. In figure 4.7, the trajectory of this meander is showed.

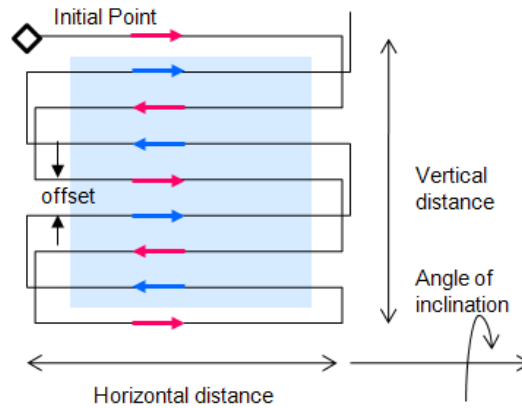


Fig. 4.7: Trajectory Meander B.

The parameters, which define the trajectory of the meander B, are the same as the meander A.

In the section 4.2.1 the meaning of the main parameters has been explained. In figure 4.8 a) four meanders, each one with a different angle of inclination (90° , 60° , 30° and 0°) are showed; meanwhile in figure 4.8 b) two meanders with different increase distance (0 and 50mm) are showed.

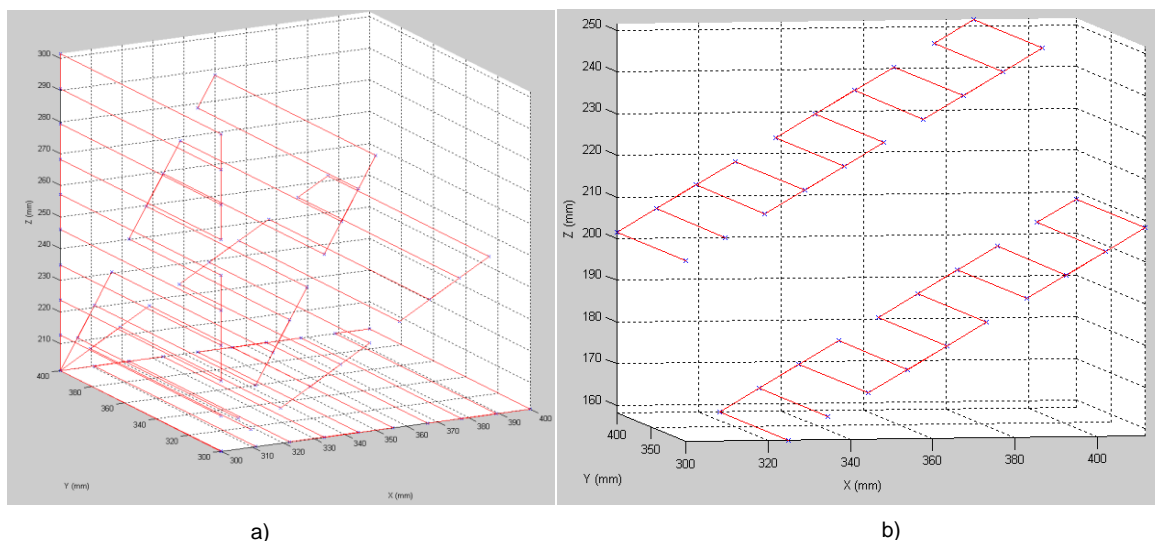


Fig. 4.8: Meander B with different angles of inclination (a) and increase distance (b).

As in the previous example (Meander A), the meander horizontal and vertical axes have been modified to make easier the translation between the systems of reference.

4.2.3 Meander C

The last kind of meander, which is going to be used for thermal spraying process, is the meander C:

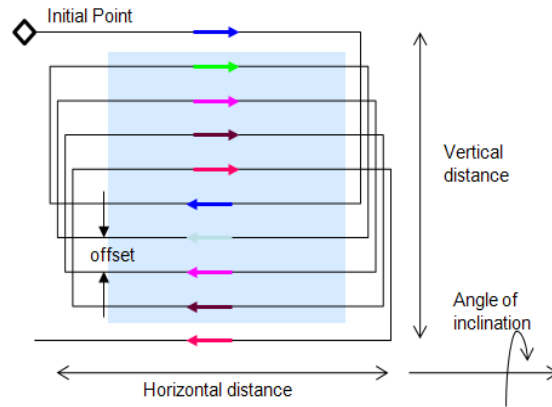


Fig. 4.9: Trajectory Meander C.

The parameters which define the trajectory are the same as in the previous meanders.

As in the previous examples; in figure 4.10 a) a graph with four meanders C, each one with a different angle of inclination (90° , 60° , 30° and 0°) is shown; meanwhile in figure 4.10 b) two meanders C, with different increase distance (0 and 50mm), are showed.

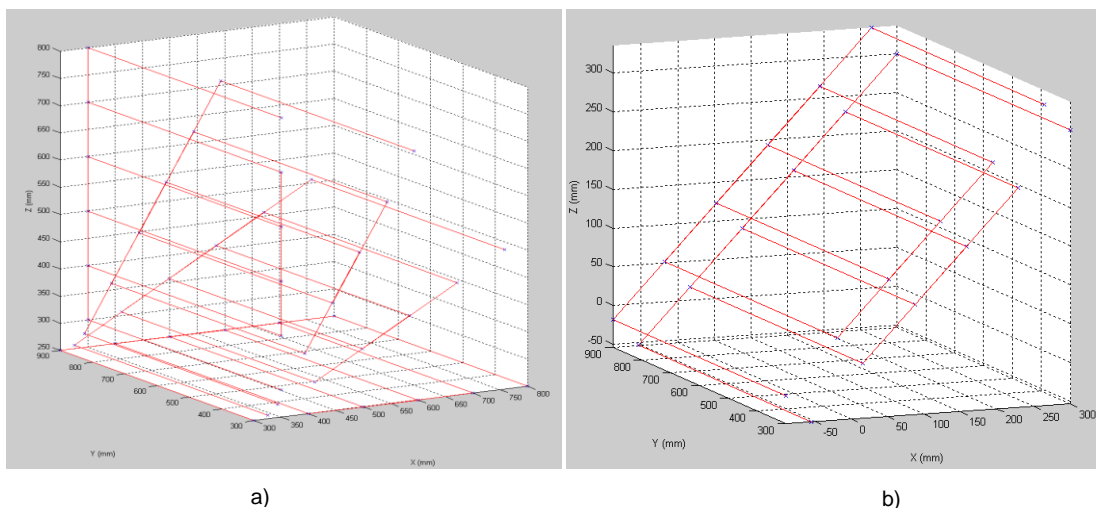


Fig. 4.10: Meander C with different angles of inclination (a) and increase distances(b).

4.2.4 Triangular (Mirror)

This trajectory has been obtained from an article [PAUL2007]. The triangular trajectory or mirror trajectory has been designed to minimize thermal variations in thermal spray processes. In this trajectory; the area, which delimits the meander, acts as a mirror. It could be seen in figure 4.11:

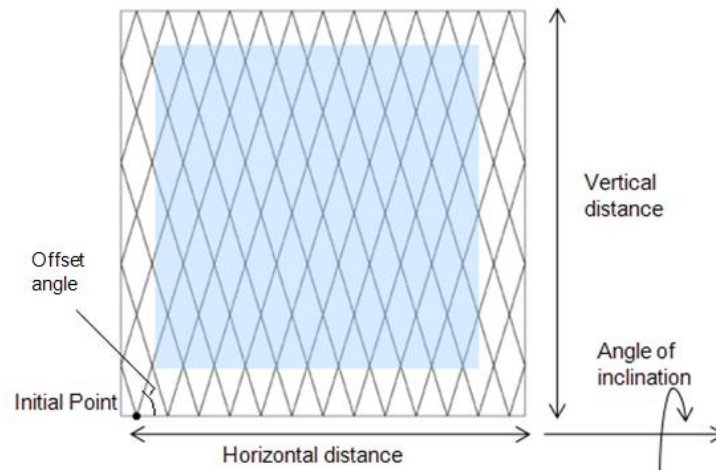


Fig. 4.11: Trajectory triangular (mirror).

The parameters, which define this trajectory, are:

- Initial Point: point in the meander horizontal axis.
- Offset angle.
- Vertical distance.
- Horizontal distance.
- Angle of inclination: angle of rotation around the axis defined by the horizontal distance.
- Increase distance: this parameter permits modified the distance of coating without modifying the first point of the trajectory.

To obtain the trajectory, showed in figure 4.11, it is necessary to calculate the initial point and offset angle optimal for each vertical and horizontal distance. In the article about this trajectory, the parameters were calculated for a specific vertical and horizontal distance. The results obtained were:

- Vertical distance = 380 mm.
- Horizontal distance = 380 mm.
- Initial Point = (13, 0)
- Offset angle = 72.89°

In this case, the system of reference is defined on the lower left corner; meanwhile in the previous trajectories, the system of reference could have been defined by the user. The system of reference is defined in that position, because it is easier to calculate the points of the trajectory. In consequence, the coordinates of the initial points are referenced to this system of reference.

This trajectory is very sensitive to changes in the parameters. In figure 4.12, two trajectories with the same horizontal and vertical distance but with different initial position and offset angle are showed. The figure a) has an initial point (13, 0) and an offset angle of 72.89°, meanwhile the figure b) has an initial point (7, 0) and an offset angle of 70°.

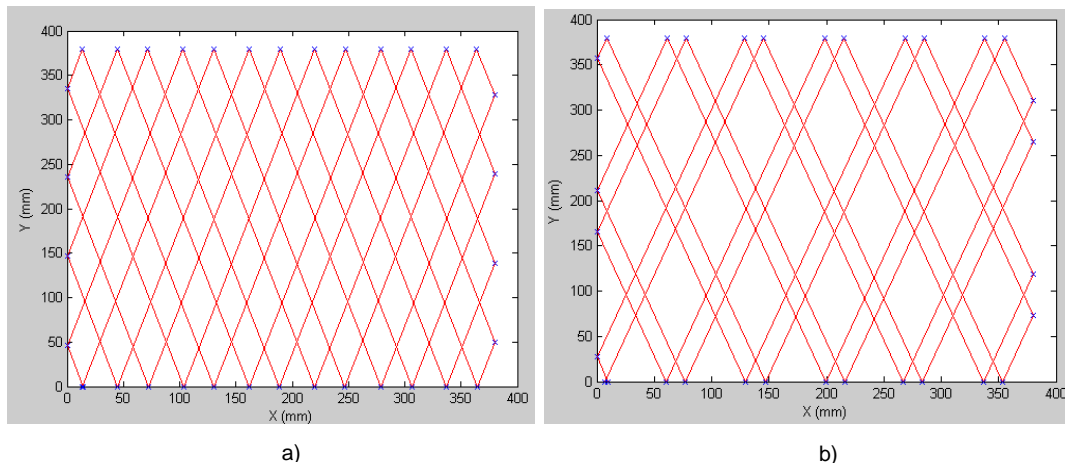


Fig. 4.12: Examples of the triangular or mirror trajectory.

In the example 4.2.1 the importance of the angle of inclination and of the increase of spray distance in the thermal and lacquer spraying processes, has been mentioned. In consequence, these parameters have been added to the program. In figure 4.13, two triangular trajectories with the optimal parameters, an angle of inclination of 45° and an increase distance of 0 and 50 mm, are showed.

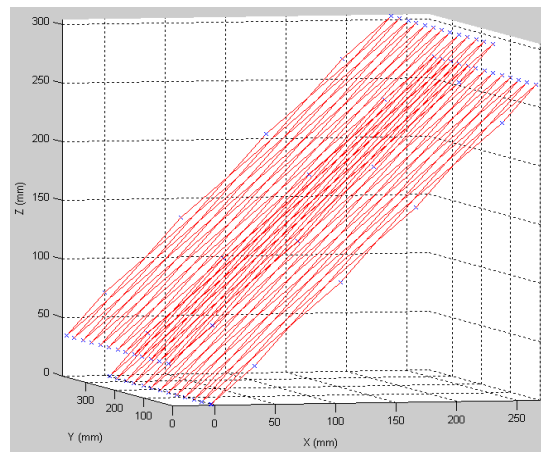


Fig. 4.13: Triangular trajectories with angle of inclination and different increase distance 0 and 50mm.

4.2.5 Variable Meander

The last trajectory, which is designed, is a variable meander. This trajectory has been designed specifically for lacquer coating operations. The main objective of this trajectory will be explained in section 4.4 in detail. The scheme of this kind of trajectory is:

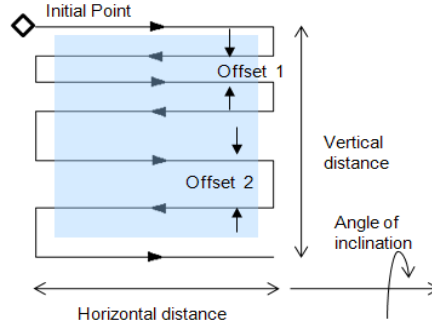


Fig. 4.14: Trajectory Variable Meander.

This trajectory is divided in two meanders type A. The first meander is going to be executed until the half of the vertical distance from the initial point; meanwhile from the half of the vertical distance until the end is going to be executed the second meander. Each meander has its own parameters. The parameters, which defined the variable meander, are:

- Initial point: point in the upper left corner.
- Vertical distance.
- Horizontal distance.
- Number of divisions 1.
- Number of divisions 2.
- Angle of inclination: angle of rotation around the axis defined by the horizontal distance.
- Increase distance: this parameter permits modified the distance of coating without modifying the first point of the trajectory.

Finally, in figure 4.15 a) three different kind of variable meander with different angles of inclination, are showed (90° , 45° and 0°); meanwhile in figure 4.15 b) the different between two increase distance (0 and 100mm) is showed.

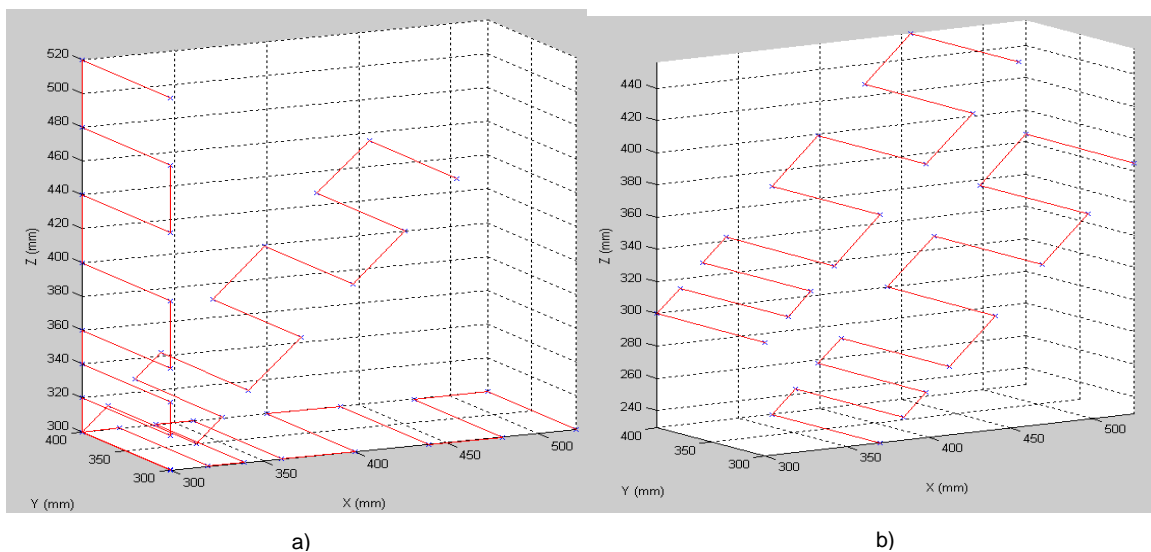


Fig. 4.15: Variable meander with different angles of inclination and increase distance.

4.3 Translation to Robot's Language

In this section, the robots and the language used are going to be described. Two kinds of robots have been used in this project. Each robot has its own language of programming, so depending on the robot the translation will be completely different.

Depending on the robot, simulation software is available or not. In consequence, the process for checking the trajectories will be in real robot or in the simulated area.

4.3.1 V+ Language



The language V+ is a programming language textual high-level developed in 1989 by Adept Technology. Nowadays, it is used mainly in industrial robots Adept marks and Stäubli robots. In the figure 4.16, a robot Stäubli Rx130, specialized in painting processes, is showed [WEBSTÄUBLI].



Fig. 4.16: Robot Stäubli Serie RX130B (Courtesy of Stäubli).

The language V+ has the following main features [MANV+]:

- Is a real time system.
- Allows continuous trajectory computation.
- Provides generation of robot-control commands.
- Callable subroutines can be used.
- Is a multitasking environment.
- Recursive, reentrant program execution is possible.

The V+ software (Adept) provides a text-based editor where commands, routines and functions are written, although any editor that creates DOS ASCII text files can be used. In this software, a simulated work area is not available; so it is just possible to check the grammar structure of the program. In consequence, the process of checking the trajectories has to be done in the real robot.

The V+ software also provides different packages for specific applications. Some of this software packages are:

- V_Trajsig: is a software package used to generate Cartesian trajectories from reference locations [MANVTRAJ].

- V_Paint: is graphic-based software for painting applications. It allows easily building programs for painting processes, managing paint parameters and following the execution of the production [MANVPAINT].

In figure 4.17 is possible to see an image of the software package V_Paint. This software package is specific for painting applications. Although the design of trajectories is not going to use this software package, it provides features, which make easier the design trajectories, without needing knowledge about language programming. Those features are: programming on-line with a graphic interface, visualize the trajectory (but not the robot and the work cell), etc.

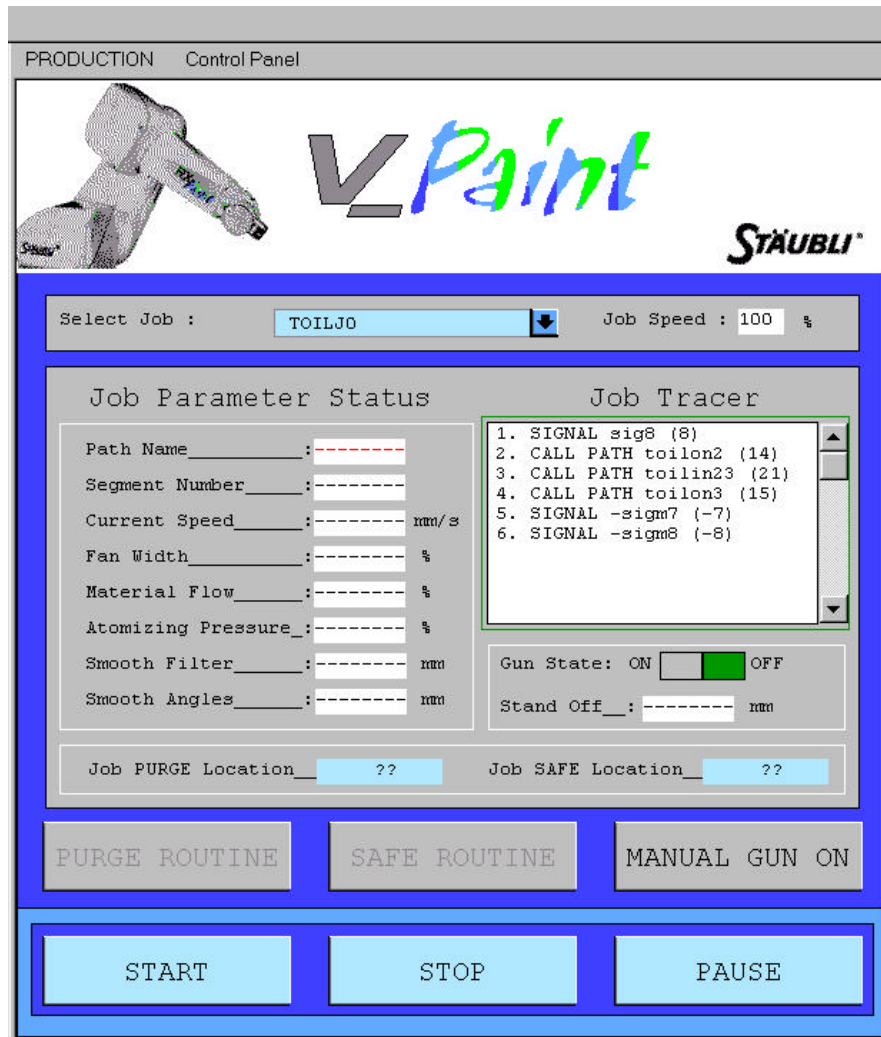


Fig. 4.17: Application of V_Paint (courtesy Adept Technology Inc.)

The grammar of this language will not be explained in this project. The information about this language can be found in the manuals, which are mentioned in the bibliography [MANVPAINT] [MANVTRAJ] [MANV+].

4.3.2 Karel Language



The Karel programming language is a practical blend of the logical, English-like features of high-level languages, such as Pascal and PL/1, and the proven factory-floor effectiveness of machine control languages. Karel incorporates structures and conventions common to high-level languages as well as features developed especially for robotics applications. These Karel features include [MANKAREL]:

- Simple and structure data types.
- Arithmetic, relational, and Boolean operators.
- Control structures for loops and selections.
- Condition handlers.
- Procedure and function routines.
- Motion control statements.
- Input and output operations.
- Multi-programming and concurrent motion support.

The software, which is used to develop the Karel program, is called “Roboguide”. This software has optional packages, which add optional features for specific applications. With Roboguide it is possible to design and test a robot system in 3-D space, so it is not necessary to use the real robot to check the trajectories, because a simulation environment is included in this software.

Roboguide is an off-line software programming. The chapter 5 will speak about the off-line software for robots applications, and this software will be described in detail in that chapter. In figure 4.18, the main interface of roboguide is showed.

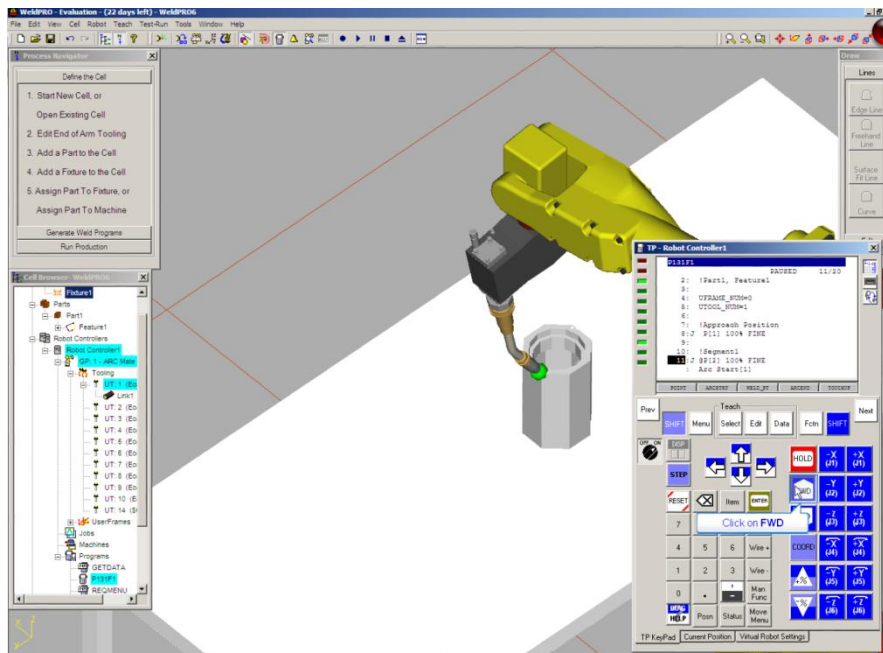


Fig. 4.18: Main interface of Roboguide with cell browser and virtual teach pendant.

As the same as in V+ language, the information about its grammar can be found in the manuals. The information about these manuals can be found in the bibliography [WEBFANUCEU] [WEBFANUCSOF] [MANKAREL].

4.4 Equipment

In this section, the different industrial robots and its purpose are described. In the IFKB, there are some industrial robots for different applications, in our case the equipments, which are going to be programmed, are:

- APS (Atmospheric Plasma Spray) cabinet.
- HVOF (High Velocity OxiFuel) cabinet.
- Lacquered cabinet.

ATMOSPHERIC PLASMA SPRAY CABINET

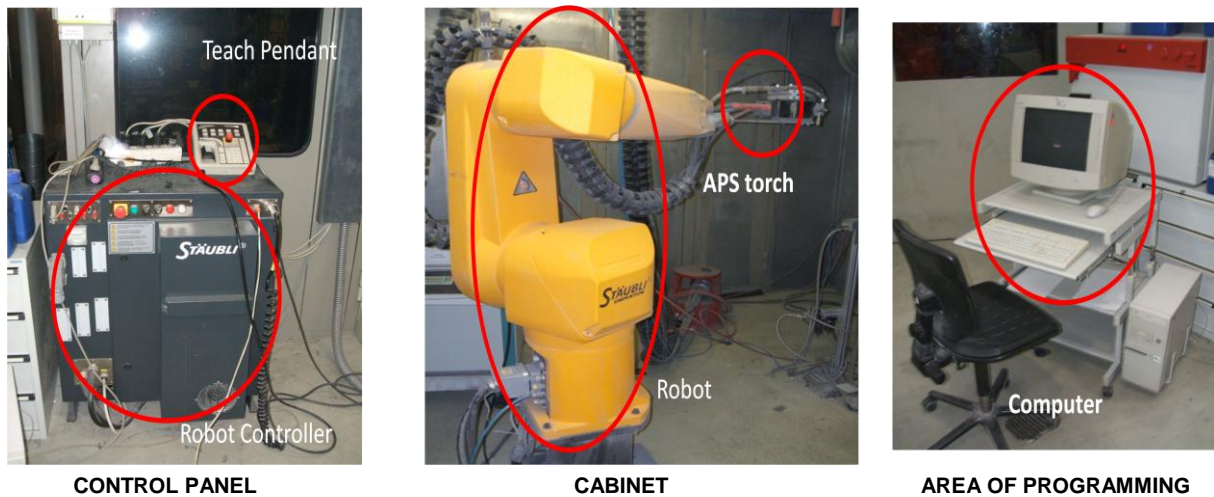


Fig: 4.19: Equipment for atmospheric plasma spraying coating at IFKB.

The control panel includes the robot controller and the teach pendant. The robot controller is from Stäubli and model CS7B. The teach pendant is included with the robot and permits to manipulate the robot manually.

The cabinet is the work area, inside of which the coating process takes place. In the cabinet there is the robot and in the extreme of the robot is the plasma gun. The robot used for plasma spraying is a Stäubli robot and model RX130C.

The last part of the equipment will be the area of programming. The area of programming is a computer near of the robot controller used to load and write program in the robot. The software used in this case is the Adept software and the programming language will be V+.

HIGH VELOCITY OXIFUEL CABINET

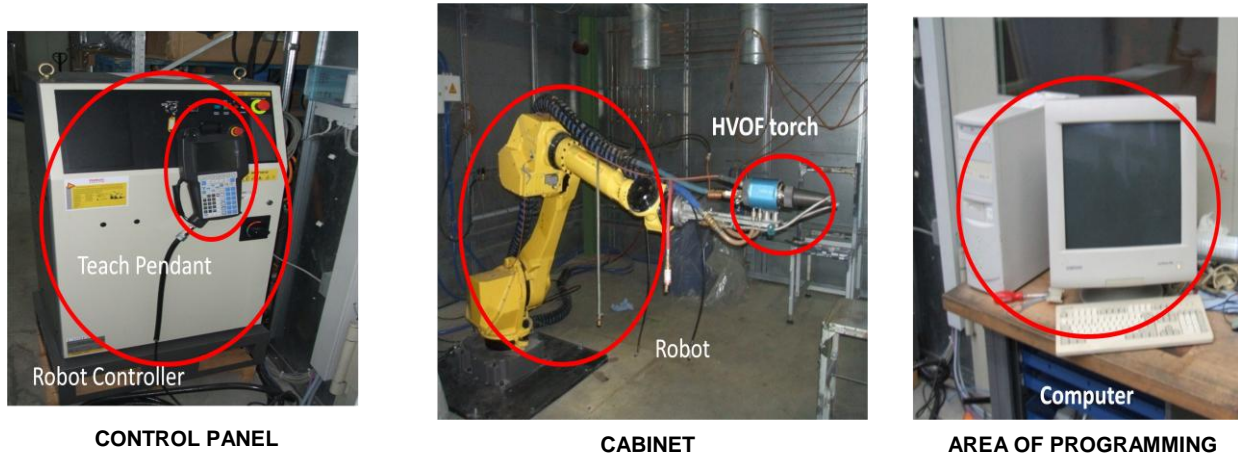


Fig 4.20: Equipment for high velocity oxifuel coating at IFKB.

The control panel is formed by the robot controller. The robot controller is from Fanuc and model R30-iA. The teach pendant is included with the robot and permit to manipulate the robot manually.

Inside of the cabinet there is the robot and in its extreme is the HVOF torch. The robot is from Fanuc and model M710iC.

The last part of the equipment will be the area of programming. The area of programming is a computer near of the robot controller used to load and write the program in the robot. The software used in this case is Roboguide and the programming language will be Karel.

LACQUERED CABINET

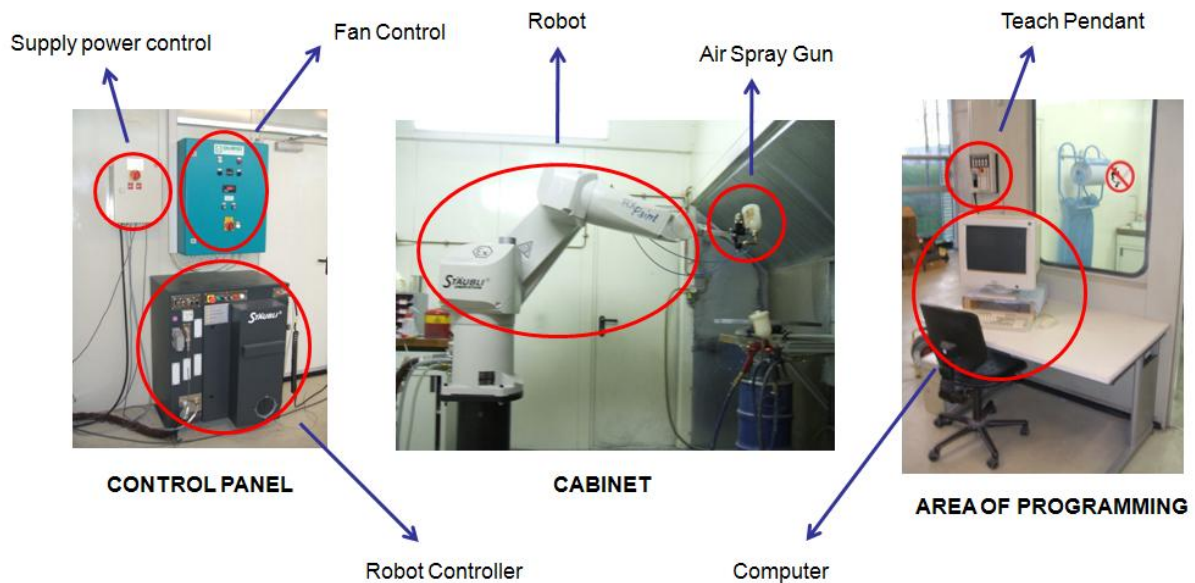


Fig. 4.21: Equipment for lacquer spraying coating at IFKB.

The control panel is formed by the robot controller, a supply power control and a fan control. The robot controller is from Stäubli and model CS7B, it is the same as in the plasma spraying equipment. The supply power control provides a security to the robot; this mechanism does not allow supplying energy to the robot controller, unless the security conditions will be met. On the other hand, a fan control permits activating and control the ventilation system.

Inside of the cabinet there is the robot and in its extreme is the air spray gun. The robot is from Stäubli and model RX130B. Finally, the area of programming is formed by the computer and the teach pendant.

The programming language is V+ and the software Adept. In this case, the lacquer process can use the specific software V_Paint of Adept. The V_Paint is specific for on-line programming and oriented to users, who do not have enough knowledge about robot programming. In this work, the simple software Adept will be used to load and write programs in the robot.

The air spray gun used for lacquered processes is from Binks and the model 95A (figure 4.22).



Fig. 4.22: Air spray gun Binks 95A.

The control of the air spray gun can be done with the robot. With some digital signals, the opening and closing of the gun can be controlled. In figure 4.23, a scheme with the connections to make automatic the opening and closing of the gun, with the signal of the robot.

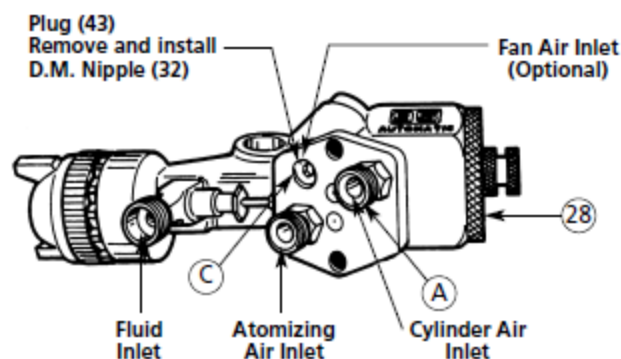


Fig. 4.23: Scheme of the air spray gun Binks 95A.

5. Off-Line Programming

In this chapter, different kinds of software related with the industrial robots are presented. This software simulates the environment and the movement of the industrial robot. The importance of this software is to simulate the robot movement with a high accuracy, without using the physical robot. In other words, one process can be optimized meanwhile the robot is working, without stopping the process of the robot.

The problem is that nowadays there are different kinds of robots from different companies (each one with its own features), and at the same time there are a lot of different kinds of software. Depending on the needs of the customer one kind of software will be selected; one objective of this chapter is going to be to know how to choose the right one. For knowing, which is the best choice in each case; a list of criteria, which permit us to classify the different software, is going to be designed.

Later, the different simulations software available in the market will be analyzed, following the list of criteria related in section 5.1. Finally, one software will be selected and it will be analyzed and tested in detail.

5.1 Criteria to evaluate Simulation Software

This section is going to make a selection of the criteria, which will be used to classify the software. The list of criteria for the software will be obtained using facilities such as related articles [CRITERIA 98] [DAVIS-EVA] [SELECT90], expert's advice, vendor's information, software manuals, and by working with some simulation packages.

It is not necessary to use all the criteria to evaluate the software. But depending on the number criteria used, the evaluation and the later decision will be more accurate.

The criteria are going to focus from three different points of view, which are showed in the next figure.

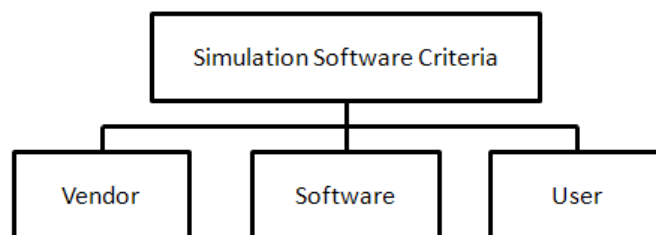


Fig. 5.1: Main criteria groups.

A short definition of each group is:

VENDOR: or company which supports the software. This group will show the facilities which the company gives to buy the software, the support in case of problem, etc.

SOFTWARE: is the main group. It is going to describe the main features of the software, its functionality, etc.

USER: describes the facilities to use the software, the experience necessary to manipulate the software, etc.

5.1.1 VENDOR

In this group of criteria, the different factors related with the vendor, which supplies the software, will be presented. These criteria are showed in the next figure.

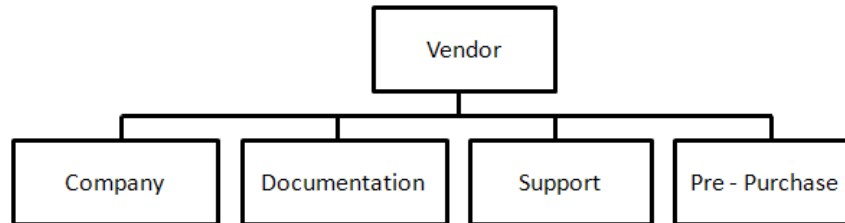


Fig. 5.2: Criteria related to Vendor Group.

As showed in figure 5.2, the group of vendor is divided in four parts: company, documentation, support and pre-purchase. With these criteria is possible to evaluate the company which sells the product and get a first idea about the software.

In the factor of company, some aspects related with the company will be evaluated. How many customers are using the product. How long it has been in the market. How successful and famous it has been. If the company is dedicated exclusively at software of simulation or it is also dedicated to other products. All these issues will give an idea of the importance of the company.

The documentation is very important in order to give a previous idea about how the software works, without the need to install the software. The documentation refers to: manuals, tutorials, examples, etc. All this information should be accurate because it will be used for solving problems and for programming.

The support is for evaluating the facilities that the company gives for solving problems. These facilities could be: training courses, maintenance of the software, update the software with the new versions, online technical support, etc.

And the last feature, which is considered, is pre-purchase. The pre-purchase is a free demo with a period of evaluation, or an on-site demonstration. It is very useful for evaluating the features of the software in a deep way, and start to familiarize with the software.

5.1.2 SOFTWARE

This group is going to evaluate the main features of the software. The criteria related with this group are showed in figure 5.3.

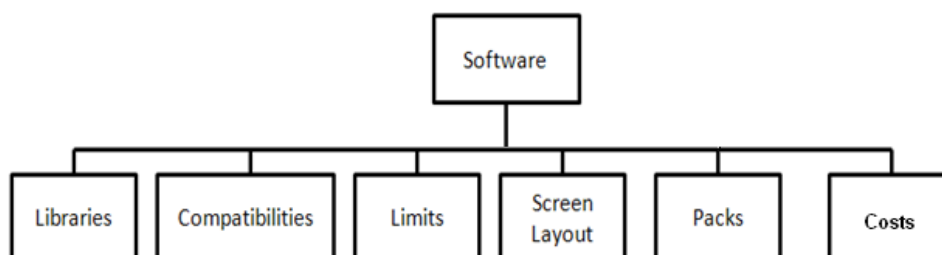


Fig: 5.3: Criteria related to Software Group.

The libraries are the components, which have the software for simulating with the robot. For example there should be libraries for: tools, objects, pieces, etc.

Another important criterion is the compatibility of the software with the operating systems and with CAX programs. In some simulation software is possible to introduce pieces made in CAX program, but these pieces should be in the right format. The format of the file is the important thing to know, for translating the piece from the design program to the simulation program.

The limits are those parameters which restrict the functionality of the simulation, some of them could be: the number of robots, kinds of robot simulated, number of paths, number of pieces, etc. One of the important limits is the kind of robot for simulating. Normally, the software permits just to simulate one brand of robot.

The screen layout deals with the issues related to the graphical environment and how the program looks.

The cost is a very important feature. Normally the simulation software is divided in packs, which add functions to the software; depending on the number of packs, the cost will be different.

The last criterion is the available packs. As told in cost, normally all the simulation softwares are divided in different packs. Some packs add functionality to the program and some of them are programmed specially to support in the development of a specific task.

5.1.3 USER

The last group of criteria is the user. The criteria related with this group are:

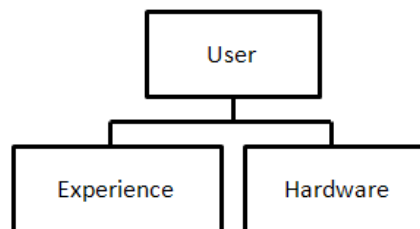


Fig. 5.4: Criteria related to User Group.

As showed in figure 5.4, this group has only two criteria. One of them is the experience; which is related with the experience necessary for using the program. Meanwhile the hardware is the equipment necessary for working with the software (pc, workstation, laptop, etc.)

5.2 Simulation software in the market

In this section, some of the simulation softwares which are on the market are described, using the previous list of criteria.

5.2.1 Stäubli Robotics Studio



The company Stäubli is a mechatronics solution provider with three dedicated divisions: Textile machinery, Connectors and Robotics. The division of robotics is the division relates Stäubli Robotics Studio; it brings together the tools necessary to program applications for robots [WEBSTÄUBLI].

In the official web page, not too much information about this software can be found. There are not tutorials about this software. In general, the information about this software in the official web page is deficient. In other words, it is necessary to contact with the company for receiving information about this product.

The company offers training courses about how to use its software. Also they have a customer support by phone. As told before, in the web page is not so much information, and it is necessary to contact with the company for getting information about the product. It is possible to get a demo version of the program and the manual of the software; with this information also examples can be found about some real applications.

The next step is taking a look at the software. In figure 5.5 the main screen of the software is showed. The screen layout is simple and not so many functionalities are showed. If some functionality has to be added, it is necessary to navigate through the software.

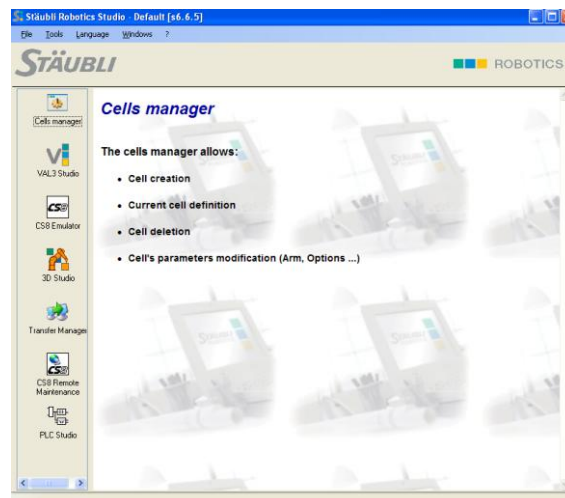


Fig. 5.5: Main screen of SRS (courtesy Stäubli Robotics).

This software is composed by 6 sub-programs, each one with its own functionality. The sub-programs which compose SRS are the next:

- VAL3 Studio: VAL3 language application editor.
- PLC Studio: Complete environment (editor, online debugger) for cell programming in PLC-IEC61131 language (Ladder, SFC, FBD, ST, IL).
- CS8 Emulator: Complete simulation of the CS8 controller and the SP1 manual control unit.

- 3-D Studio: 3D robot viewer for trajectory development.
- Downloading Tool: Transfer applications between the PC and the controller (VAL3 or PLC), easily managing all data necessary for the cell.
- Remote Maintenance: Remote display and action on the CS8 controller for maintenance operations [MANSRS].

This software operates with Windows XP and it has been tested with the version Windows XP SP2. The minimum recommended hardware for SRS is a PC Pentium 1.5 GHz (or equivalent) and 512 Mb of memory [MANSRS]. If all these requirements are completed, the software could be installed in the computer. Then it is not necessary to have a workstation for working, with a simple laptop which completes the requirements, it is enough.

The software is provided with a library of robots. But no library of components, pieces, tools, etc. is supplied; because with this software only the movement of the robot can be simulated, but it cannot simulate the environment. Another disadvantage is that the software is limited to simulate only the robots of the Stäubli Company.

Finally, there is a pack specifically designed for painting process, this pack is called PAINTIXEN. The PAINTIXEN software package is designed to simplify the use and programming of the applications of conventional or electrostatic powder and liquid paints. It is integrated in the Stäubli Robotics Studio software [WEBSTÄUBLI].



Fig. 5.6: Screen of Paintixen program (courtesy Stäubli Robotics).

The main features of Paintixen software can be resumed in:

- Software tool for the flow control. This feature allows to measure the quantity of paint used on the last cycle.
- Circle function makes it possible to generate curves and circles.
- Swing function at entry and exit of trajectory; this function allows the optimization of parts quality and their finish, particularly on cross layers.
- Plane function sets points parameters to easily generate trajectories to cover flat surfaces. These last 3 functions can be linked to obtain optimum paint quality [MANPX].

5.2.2 Roboguide (Fanuc)



FANUC is a Japanese electromechanical manufacturer specializing in robotics. FANUC had its beginnings as part of Fujitsu developing numerical controls (NCs) and servo systems. The company name is an acronym for Fuji Automatic Numerical Control [WEBFANUCJP].

The simulation software is called Roboguide. Information about this software can be found in the official web page [WEBFANUCEU]. As in Stäubli software, manuals cannot be found in the web page, but tutorials and videos can be found in the official web page. These tutorials and videos are very useful, when the programmer is a beginner in this software. But there is not demo version of the software in official web page, so it is necessary to contact with the company for getting a demo.

The supports offered by Fanuc consist on a call center and also in a service of repair. Also the company offers training courses in their offices.

In figure 5.7, the main screen is showed with a new cell with a robot. The main functions can be found with a simple look. Also the teach pendant is simulated in other windows, which is very useful to program the robot.

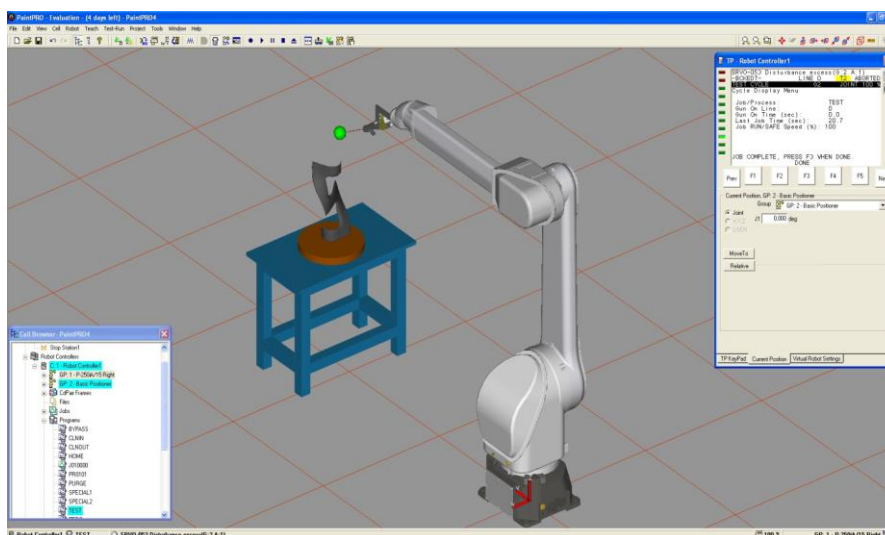


Fig. 5.7: Main screen of Roboguide (courtesy Fanuc robotics).

This software offers different libraries for tools, robots, pieces, etc. The inconvenience is that with this software it is only possible to simulate robots of the same brand as the software, Fanuc Robots. On the other hand, piece designed by other programs (CAX programs) can be imported to this software. The file of the piece has to be in IGES format [ROBDAT09].

The big difference between this software and the Stäubli software is that in this software all the environment of the robot can be simulated. So, it is possible to detect if the robot is going to have some collision with some object around it. Meanwhile with Stäubli software the robot and its movement were only simulated.

The software is compatible with the next operating systems: Windows XP (32bit, 64bit) and Windows Vista (32bit, 64bit). Also the computer has to be installed DirectX 8.0 and internet explorer 6.0 or later version. The features of the computer must be at least: a Pentium IV with 1.5 GHz, 500Mb of memory RAM, and 2GB hard drive disk space [ROBDAT09].

The Fanuc Company offers a number of software packages specifically designed for some tasks. These software packages are the next:

- **Handling PRO:** it is a member of the ROBOGUIDE family of offline robot simulation software products built on the Virtual Robot Controller. Handling PRO allows users to simulate a robotic process in 3-D space or conduct feasibility studies for robotic applications without the physical need and expense of a prototype work cell setup.
- **PaintPRO:** this software is a graphical offline programming solution that simplifies robotic path teaches and paint process development.
- **PalletPRO and PalletTool:** PalletPRO and PalletTool palletizing simulation package helps save time and expense for the development and integration of robotic palletizing and depalletizing systems. ROBOGUIDE-PalletPRO and PalletTool is built on the Virtual Robot Controller and is a plug-in option to ROBOGUIDE simulation software. It is also available as a stand-alone package for palletizing integrators and end users.
- **WeldPRO:** latest plug-in to the ROBOGUIDE off-line programming tool, allows users to simulate a robotic arc welding process in 3-D space. Driven exclusively by a FANUC Robotics Virtual Robot Controller, WeldPRO is empowered with the most accurate program teaching tools and cycle time information available in any simulation package [WEBFANUCSOF].

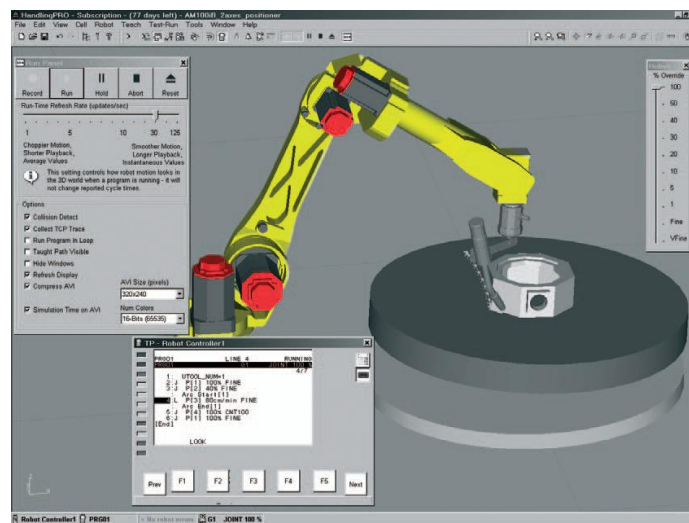


Fig 5.8: Screen of WeldPro program (courtesy Fanuc Robotics)

5.2.3 Simpro (Kuka)

The company name, KUKA, is an acronym for Keller Und Knappich Augsburg and at the same time is the registered trademark found on the industrial robots and other products they produce. KUKA Robotics produces industrial robots for a variety of industries: from automotive and fabricated metals to food and plastics. The company headquarters are located in Augsburg, Germany [WEBKUKACOMP].

The simulation software made by Kuka is called SIMPRO. In the official web page, a lot of information can be found about this program. There is also a quick guide about SimPro; one of the differences with the previous software is that a demo version of the program can be downloaded from the official web page. Also libraries of pieces can be downloaded from the web page and they can be used with the demo [WEBKUKASOF].

The company has a technical support, in this aspect is the same that in the last two companies. Also the company offers different kind of training courses. The training courses should be done in the branch office.

The software is compatible with the next operating systems: Windows 2000 and Windows XP (32bit). The features of the computer must be at least: a Pentium III with 1.0GHz, 256Mb of memory RAM. The resolution of the screen should be at least 1024x768 and 16-bit color. An Open-GL graphics card is recommended to improve the view of the simulation [QUICK_KUKA].

The main screen of the software is very intuitive and friendly (figure 5.9). In this case, it is not possible to simulate the teach pendant of the robot. But it is possible to simulate the environment around the robot.

The software is provided with different libraries about tools, elements, robots, etc. The problem, as for Roboguide, is that only robots from Kuka Company can be simulated. The libraries about pieces, tools, etc. can be incremented adding piece designed by other programs. The software permits transfer 3D piece developed in other programs (CAX programs), the only requirement is that the file has to be in IGES format.

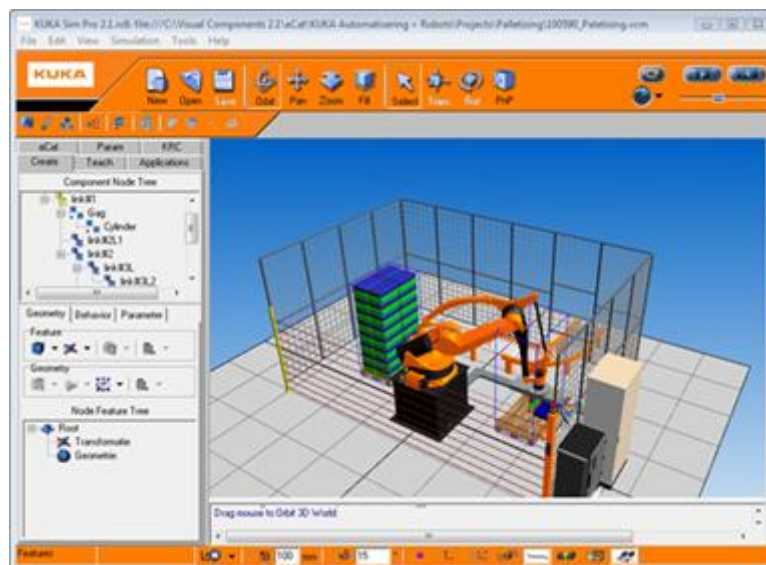
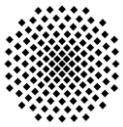


Fig. 5.9: Screen of Simpro (courtesy of Kuka Robotics Company).



The software Simpro has also different packs, which add new functions to the base software. These packs are not oriented to specific applications, like in the case of Roboguide or Stäubli; in this case, it is oriented to develop the environment of the robot and the possibility to work and connect with the real robot. The packs can be resumed in [QUICK_KUKA]:

- **KUKA.Sim Viewer:** for optimal presentation of the simulation.
- **KUKA.Sim Layout:** program for creating 3D lay-outs of systems with KUKA robots, and for simulation and workflow investigation.
- **KUKA.Sim Pro:** for the offline programming of KUKA robots. The product is connected in real time to the virtual KUKA controller, thus allowing cycle time analyses and the generation of robot programs.

5.2.4 Famos (Easy Rob 3D)

The Easy-Rob™ Simulation System has been on the market since 1996 and the company headquarters is located in Frankfurt am Main. Between all the software, which this company provides, the Famos software is the software specialized in off-line programming [WEBEASY].

Famos is an offline programming system. The software package Famos enables the efficient and simple programming of industrial robots of many brands. In the official web page much information about the software could be found, like manuals, tutorials and some examples. A free demo can be downloaded from the official web page.

The supports offered by Easy-Rob consist on a call center, a maintenance service (software updates, technical support by email, etc.) and training programs. The training programs can be conducted in-house or at the Easy-Rob headquarters in Frankfurt.

The main screen of the program is showed in figure 5.10. The software is very friendly and with very easy access to the functions. It is possible to simulate the environment around the robot. In this software, there are not available libraries about piece, tools, elements, etc. but it is possible to simulate robots from different brands, some of them are: adept, Kawasaki, Unimation, IGM, Fanuc, Kuka, Stäubli, etc.

There are some libraries of robots, which need a special license. It is the case for example of: Fanuc, Kuka, Stäubli, etc. Without the license these kind of robots can be moved only by axes.

There are not libraries about pieces, tools, etc. But it is possible to transfer CAD files designed by other programs (CAX programs), with this feature persolalized libraries can be designed. Famos can process files of common CAD-Systems on the market. The standard exchange file formats STEP(AP203 and AP214), VDAFS and IGES as well as the ACIS SAT and Parasolid XT files formats are supported. ProENGINEER® and Wildfire® files can be used natively and without loss [WEBEASY].

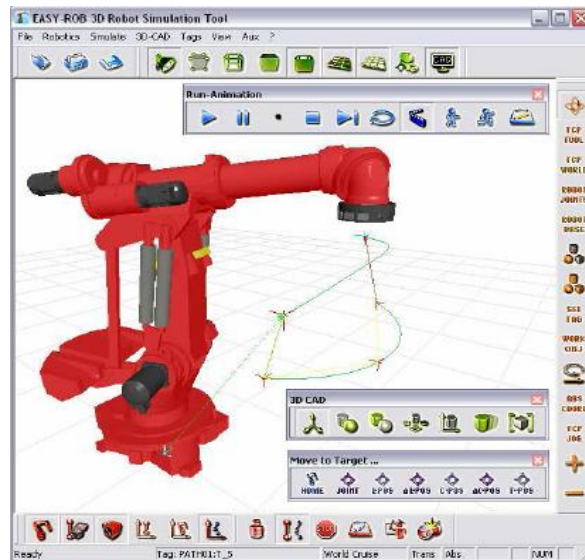


Fig. 5.10: Screen of Famos (Courtesy of Easy Rob 3D).

Famos executes on an IBM compatible Pentium PCs with the next operating systems: Windows XP, Windows Vista Business and Windows 7 Professional, Ultimate & Enterprise. The memory has to have at least 1024 MB of RAM installed (2048 MB RAM is recommended). Graphic acceleration cards can really boost application performance. The nVIDIA GeForce or ATI graphics cpu with the OpenGL 2.0 driver installed and 256 MB VRAM are recommended.

5.3 Roboguide (Fanuc)

The software roboguide is the software used in the IKFB. In this chapter, the features about the software and the main functions, which can offer this software, will be explained in detail.

5.3.1 Main Features

First, a description of the software and the packages, which are going to be used, will be given. In section 5.2.2, the main features and the packages, which compose Roboguide, were explained. The main features of Roboguide can be resumed in:

- Cad-to-Path Programming.
- The FANUC robot library.
- CAD Import (IGES files).
- Simple robot teaching.
- Collision detection.
- TP Trace.
- Animation AVI.
- Virtual Teach Pendant.
- Etc.

The feature Cad-to-Path programming consists on design the trajectory of the robot, taking into consideration the CAD piece. In other words, the trajectory will follow the shape of the CAD piece. The software has included a number of predefined trajectories (figure 5.11). Once the trajectory has been selected and set up on the piece, the software calculates the points of the trajectory automatically.

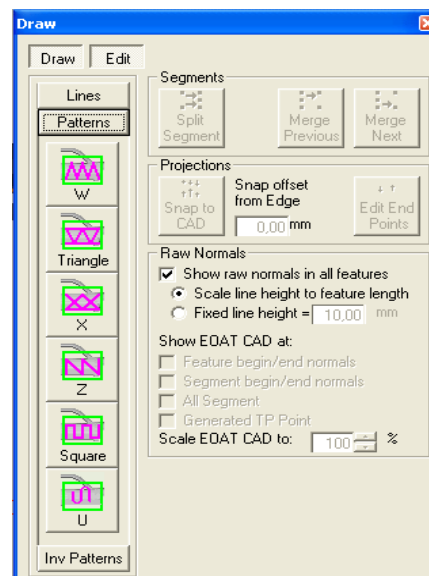


Fig. 5.11: Interface with the possible patterns trajectories in Roboguide.

The software includes a library with all kinds of robots from Fanuc. Also, it includes a number of libraries about tools, pieces and elements; which will help to realize the design of a virtual cell.

The collision detection permits to see the possible collisions, which can have the robot with the elements, which are around it. The TP trace permits to visualize the trajectory

of robot after the simulation (figure 5.12); while videos in AVI format can be recorded during the simulation.

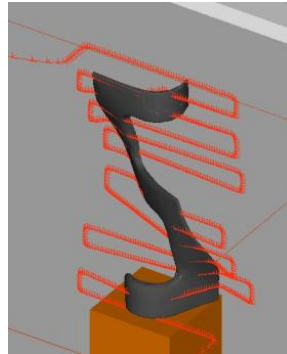


Fig. 5.12: Trajectory followed by the robot in a coating process.

Finally, the teach pendant is simulated in this software, which permits to control the robot more intuitive and similar to reality (figure 5.13).

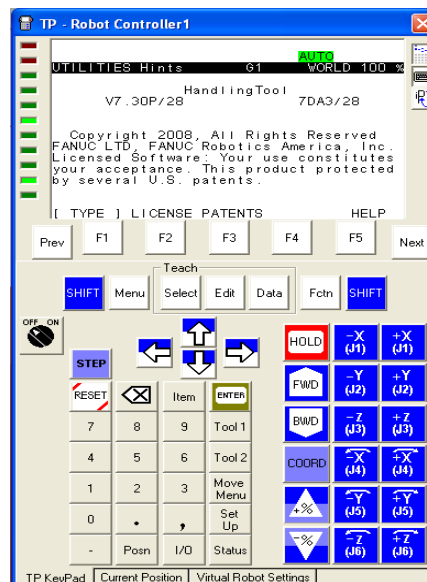


Fig. 5.13: Virtual Teach Pendant in the software Roboguide.

5.3.2 Trajectory Generation

As saw in chapter 4, the design of trajectories plays an important role in the thermal spraying processes. In this section, the different ways to create a trajectory with Roboguide will be analyzed.

First, a virtual cell is going to be created. The virtual cell simulates the work environment, where the robot Fanuc M710iC is installed in the IFKB. The real cell has been measured and with these measures the virtual cell has been designed (figure 5.15).

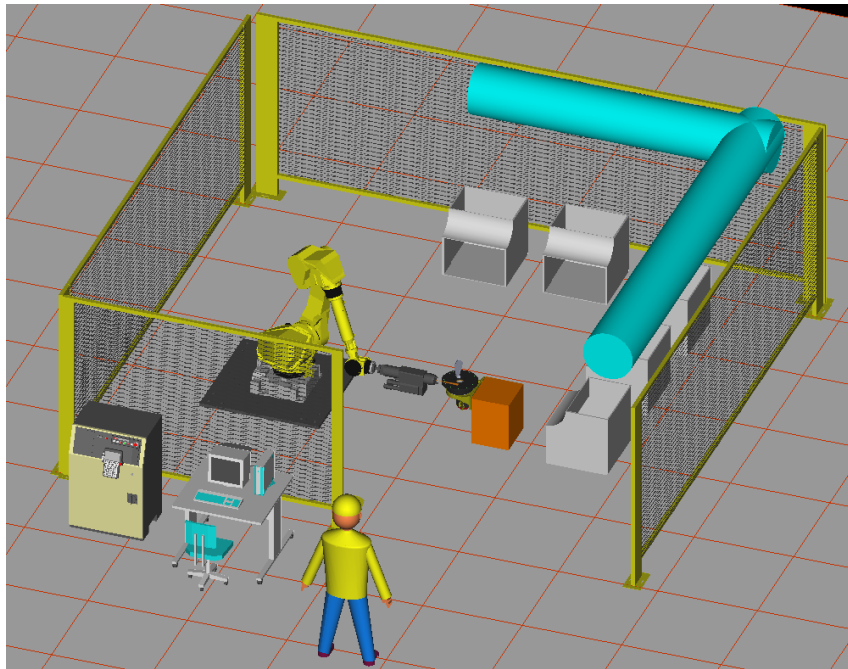


Fig. 5.15: Virtual cell with the robot M710iC.

The different ways to design a trajectory in Roboguide are:

- Point to Point programming.
- Cad-to-Path programming.
- Karel programming.

Trajectory 1 – Point to Point

The easier way to design a trajectory is introducing the points manually. Roboguide permits to introduce the points manually, moving the robot with the teach pendant and recording the points, which have to follow the robot. Also it provides an interface, which permits to reorder the points and associate actions in each point, for example open hand, close hand, activate gun, etc. In figure 5.16, an example with a trajectory and the mentioned interface are showed.

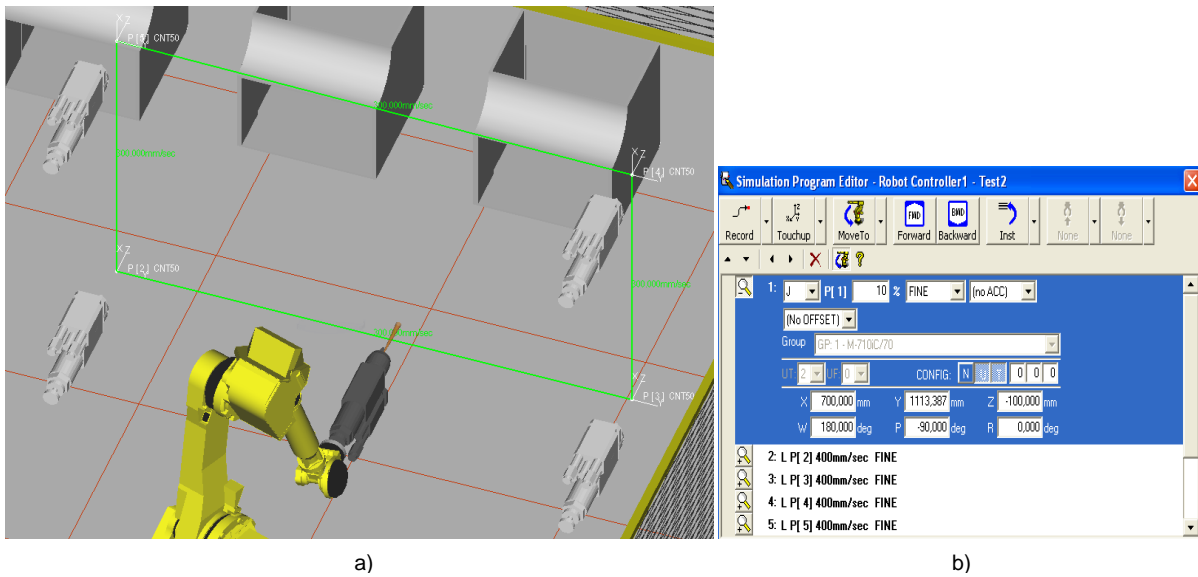


Fig. 5.16: Trajectory designed point to point (a) and the interface to introduce the points (b).

Trajectory 2 – CAD to Path

The feature CAD-to-Path is a way to design trajectories, taking into consideration the surface of the piece. A piece can be introduced in CAD format (IGES format) and specify a trajectory in this piece. The different patterns trajectories, which can be done, are in figure 5.12. Once the trajectory has been chosen and set up, Roboguide calculates the points of the trajectory. In figure 5.17, a piece with meander trajectory is shown.

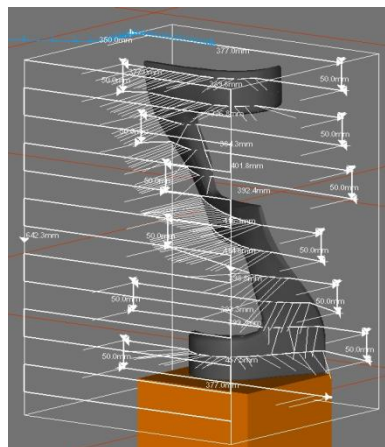


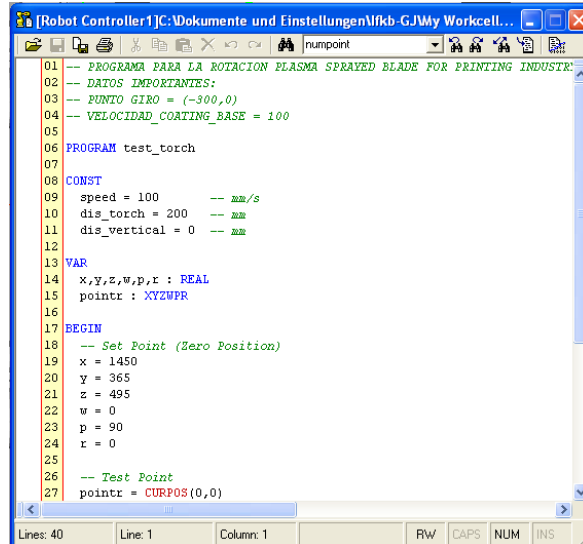
Fig. 5.17: Piece in Roboguide with a meander trajectory.

One of the advantages of this programming is that the normal vectors on the surface of the piece are calculated automatically. As saw in chapter one, the spray angle and spray distance are a very important parameter in the thermal spraying process, with this feature is possible to keep constant this parameter through the entire trajectory. This trajectory programming offers also an interface, which provides many options to set parameters of the trajectory (offset meander, initial point, etc.).

In resume, this feature offers a great advantage to design trajectories in thermal spraying process. Because it calculates the associated normal vectors and develop the trajectory taking this vectors; which permits to keep constant the thermal spraying parameters on the robot.

Trajectory 3 – Karel Program

The last way to design or program a trajectory in Roboguide is through a Karel program. Karel is a programming language used to robots Fanuc. Roboguide provides a text editor to develop Karel programs and also this text editor serves to correct mistakes in program syntax. In figure 5.18, the text editor for developing programs in Karel is showed.



```
01 -- PROGRAMA PARA LA ROTACION PLASMA SPRAYED BLADE FOR PRINTING INDUSTRY
02 -- DATOS IMPORTANTES:
03 -- PUNTO GIRO = (-300,0)
04 -- VELOCIDAD_CORTING_BASE = 100
05
06 PROGRAM test_torch
07
08 CONST
09 speed = 100 -- mm/s
10 dis_torch = 200 -- mm
11 dis_vertical = 0 -- mm
12
13 VAR
14 x,y,z,w,p,r : REAL
15 pointr : XYZWPR
16
17 BEGIN
18 -- Set Point (Zero Position)
19 x = 1450
20 y = 365
21 z = 495
22 w = 0
23 p = 90
24 r = 0
25
26 -- Test Point
27 pointr = CURPOS(0,0)
```

Fig. 5.18: Text editor for language Karel.

6. Implementation of an external axis.

In this chapter, the knowledge, which has been seen in chapters 4 and 5, is going to be used for the implementation of an external axis.

The pieces, which are coated in thermal spraying processes, can have different shapes. The angle of coating is very important, as told in section 1.3; this angle should be perpendicular to the surface of the piece (90°). For keeping this angle constant, it is necessary that the robot moves following the shape of the piece. Depending on the shape of the piece, the trajectory, which has to follow the robot, will be more or less complicated. When the piece has a shape, which is really complicated, the trajectory is really difficult to calculate or even impossible to perform.

In those cases, coating trajectories are impossible to perform with the piece in static; it is necessary to implement an external axis. The external axis permits that the piece rotates in one axis. With this new degree of freedom it is possible to access with the robot parts of the piece, which in static would not have been possible. In figure 6.1, a piece with a complex geometry, is showed.

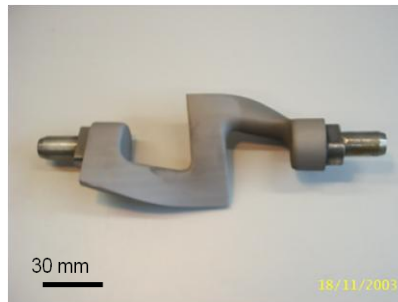


Fig. 6.1: Sigma-form blade for injection molding application.

The use of an external axis introduces more degrees of freedom to design the robot trajectory. The calculation of the trajectory has different solutions depending on many factors. The approaches, which have been taken to solve this problem, are explained in the sections 6.1 and 6.2. There are more possible ways to calculate the trajectory of the robot and the external axis; but it will be proposed as further works.

6.1 Approach I

6.1.1 Concept

The concept of this approach consists in simplifying the movement of the gun. The gun can move only along two axes (X and Y), while the rest of the coordinates, which are related with the orientation of the robot (yaw, pitch and roll), are kept constant.

First, the normal vectors of the piece are calculated in each point of the trajectory (figure 6.2). As told, the gun should be perpendicular to the surface, to achieve this purpose the piece is going to rotate.

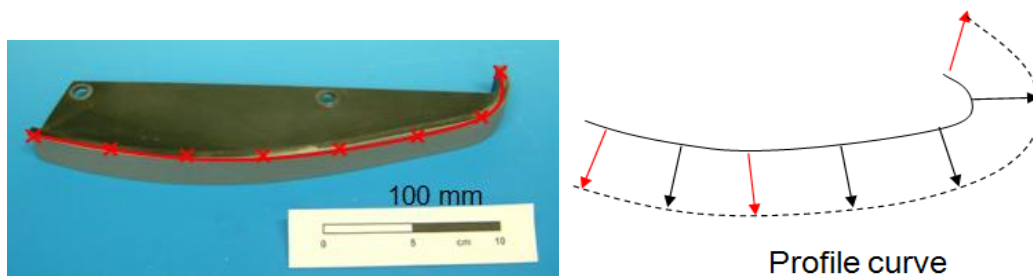


Fig. 6.2: Profile curve with normal vectors from a blade for printing industry.

The rotation of the piece will be coordinated with the robot to find in each instant the robot position and external axis position, in which the gun is perpendicular to the surface of the piece. In figure 6.3, a scheme with the rotation of the piece in three moments of the trajectory, which the robot should execute, is showed.

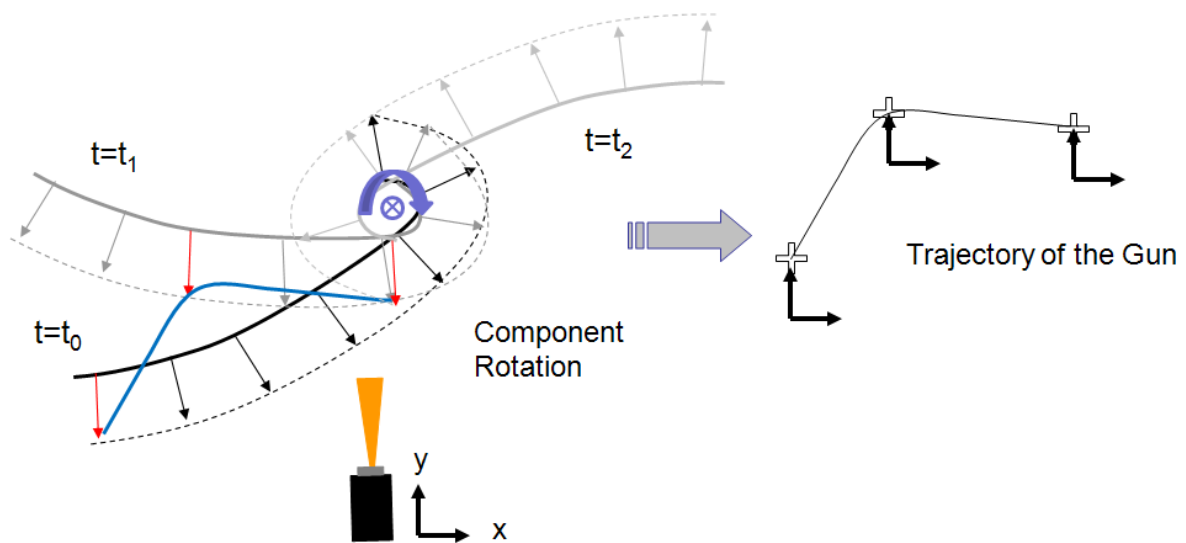


Fig. 6.3: Scheme of the approach I.

With this aim, the external axis rotates until the next point of the substrate which is considered for the trajectory, in a way that the normal vector of the substrate at that point has the same direction as the torch longitudinal axis. At the same time, the robot moves in the XY plane to reach its position in which it will be aligned with the normal vector of the substrate surface at the corresponding point. With this coordination, it is possible to keep constant the parameters: spray distance and spray angle.

Another parameter, which is very important in thermal spraying process, is the coating velocity. This velocity should be constant in all the process. In consequence the velocities of the robot and the external axis in each moment have to be coordinated to achieve a constant coating velocity. The calculation of the velocity in each instant of the trajectory is based on the “relative velocity between two particles”; explained in the appendix 10.1.

In this approach, it is possible to manipulate one parameter, which is going to modify the trajectory of the robot. This parameter is the piece rotation center; depending on the position of the piece on the external axis, the trajectory and the velocities will be completely different. In the examples of section 6.1.3, the consequences of changing this point, will be explained better.

6.1.2 Steps to design the trajectory

The trajectory of the robot has to be calculated in consequence with the shape of the piece.

The programs, which have been used to design the trajectory, are: Matlab and Roboguide. Matlab is used to make all the calculation; meanwhile Roboguide is used just to test and verify that the trajectories are right. In figure 6.4, the general scheme with the steps followed to design a trajectory, is showed.

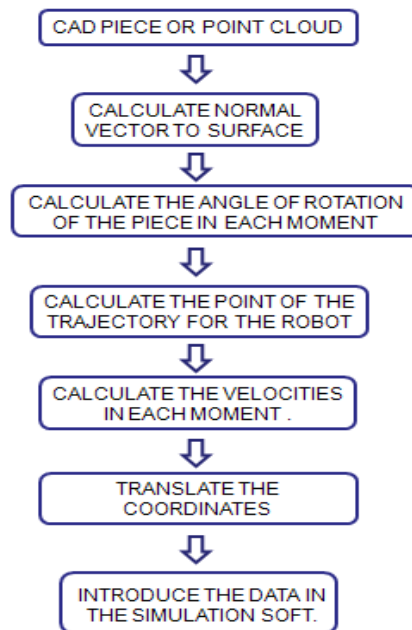


Fig. 6.4: Steps to design a trajectory with an external axis.

The first step is to obtain the points of the piece, which are going to be coated. It can be possible to obtain with the CAD piece or directly with a point cloud. Once the points have been obtained, the normal vectors of the surface are calculated.

With the points of the piece and the normal associated vectors to each point; it is possible to calculate the trajectory, which has to follow the robot, and the positions of the external axis. The trajectory, which is going to be followed by the robot, will be a Meander A; a scheme about Meander A has been showed in the section 4.2.1. Finally the last step realized in Matlab will be the calculation of the velocities; this calculation can be realized without needing to translate the coordinates to the system of reference of the robot.

At the end, all the data obtained in Matlab will be translated to the Roboguide and introduced in a program. The translation of the trajectory from Matlab to Roboguide has been seen in chapter 4, and the language of the Roboguide is Karel. Also the Karel language provides a function, which permits to make the change of system of reference without needing calculation in Matlab. Therefore, the translation of coordinates from the matlab reference system to the real robot reference system will be made in Roboguide.

This scheme is general and, depending of the geometry and the file format of the piece, the program, which obtains the points of the piece, will be more or less complicated. Different programs have been designed to obtain these points and its normal associated vectors, depending on the piece geometry and available data.

6.1.3 Examples

In this section, some pieces have been tested. Each example has its own purpose, from testing the position of the rotation center to how to obtain points from a complex geometry. Five examples are exposed:

- Blade formed by two arcs with rotation center in position 1.
- Blade formed by two arcs with rotation center in position 2.
- Airfoil NACA 5020.
- Sigma for injection molding application (Knetter).
- Blade for printing industry (KBA).

With the two first examples it will be possible to see the differences, which are produced in the trajectory and velocities, changing the rotation center. The Airfoil example will show us the limitations, which can have the external axis. The blade for injection molding application has a very complex geometry; in this case a special program to obtain points will be developed. Finally with the blade for printing industry, the process from a real piece, until the final trajectory with the real robot will be developed.

The results obtained from the execution of the programs and from the simulation are showed here.

EXAMPLE 1 – Blade formed by two arcs with rotation center in position 1

The first example is an easy piece. The profile of this piece is composed by two arcs, which are tangent in one point; and later this profile has been extruded to generate the surface of the piece. The design of this piece has been made in Matlab with a program, which calculates mathematically the points of the piece without starting from a CAD file or a cloud of points.

The piece has to be introduced also to the simulation software. To get the piece in CAD file, the points have been translated to a CAX program, to make the piece in an IGES format; which will be introduced in the simulation software to test the trajectory. In figure 6.5, the piece in CAD is showed.

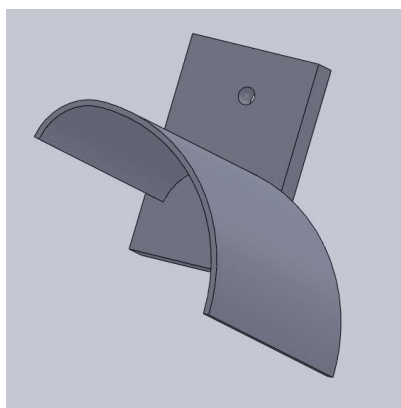


Fig. 6.5: Blade formed by two arcs in CAD format.

For each point of the piece, the associated normal vectors have to be calculated. In figure 6.6, the two arcs piece with its associated normal vectors to each point is showed.

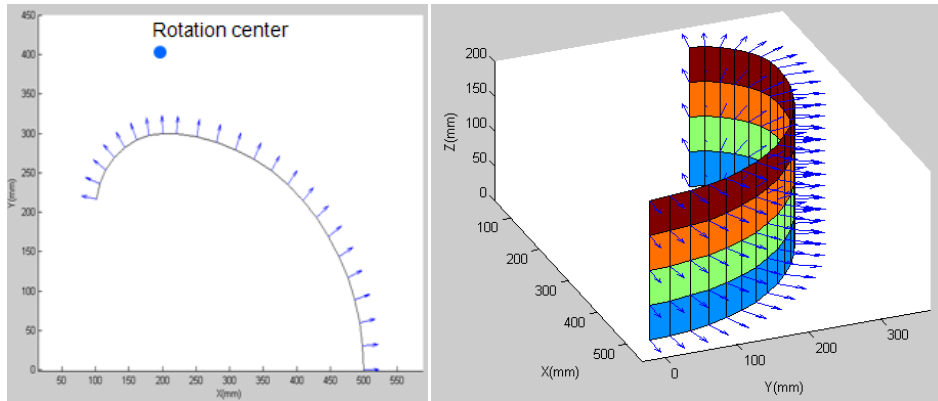


Fig. 6.6: Blade formed by two arcs with its normal associated vectors.

The next step, in the design of the trajectory is to calculate the angle of rotation of the piece for each moment. This angle of rotation will be the angle necessary in order to locate the next point of the substrate surface which is considered in the trajectory, perpendicular to the torch. In figure 6.7, a graph with the respective angles positions, which have to have the external axis in each moment, is showed.

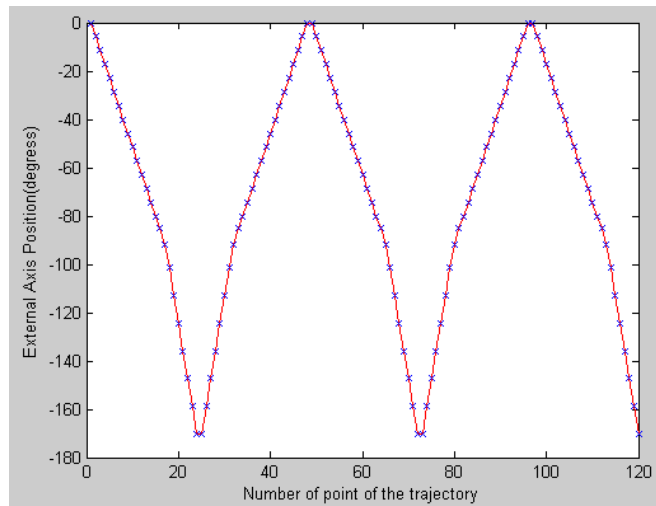


Fig. 6.7: Graph with the angular position in each instant of the external axis.

The angles that the piece has to rotate to reach each position in the trajectory are not influenced by the location of the rotation center. So, independently of the situation of the rotation center, the angle, which has to rotate, will be the same. In figure 6.7, it is possible to see symmetry in the graph; this is because the profile of the piece is extruded along the Z axis. Then the angle, which has to rotate the piece to be perpendicular to the focus of the gun, will be the same for the points with same X and Y coordinates.

The trajectory of the robot will be calculated with the data obtained until now. The center of rotation is configured at the position (200,400), in the system of reference of Matlab. In figure 6.8, the trajectory of the robot with the different positions of the piece and without the piece is showed.

The last calculation is to obtain the velocities. The calculation of the velocities have been based on the “Relative velocity between two particles”, this concept is explained in the appendix 10.1 in detail. In figure 6.9 a) a graph with the velocities of the robot in each moment is showed; meanwhile in figure 6.9 b) a graph with the angular velocities of the external axis in each instant can be seen.

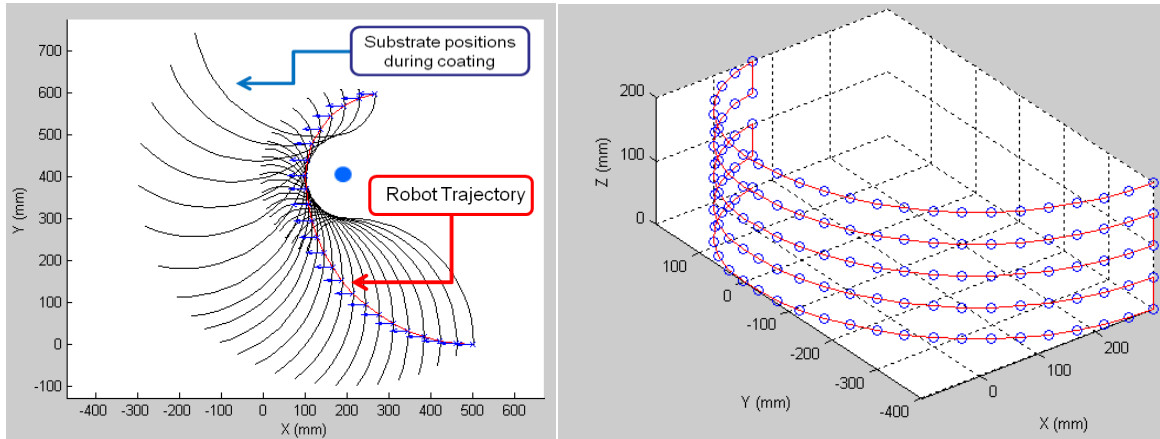


Fig. 6.8: Graphs with the trajectory of the robot.

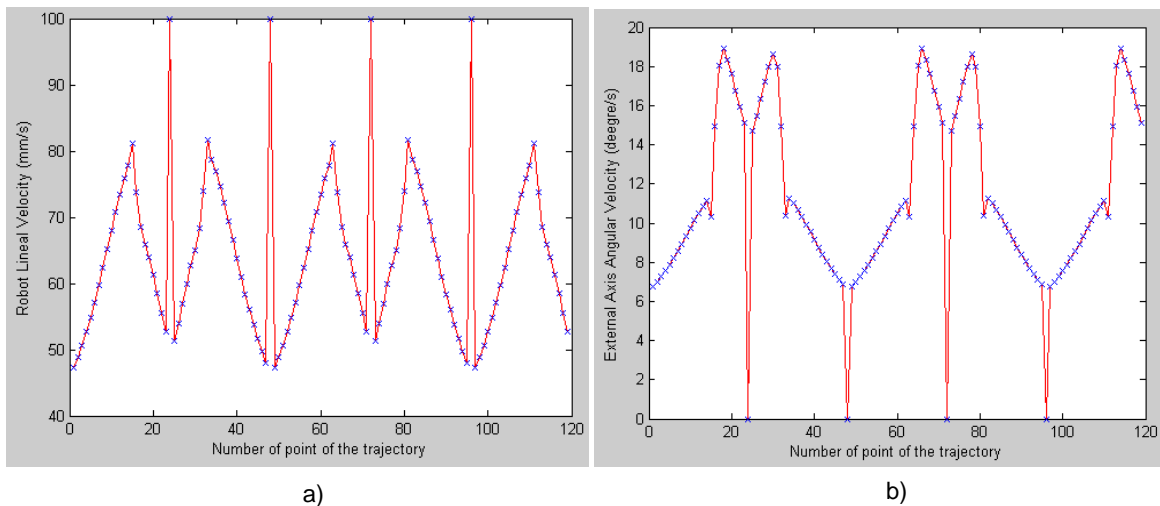


Fig. 6.9: Graphs with the velocities of the robot (a) and the external axis (b).

It is possible to see the symmetry in the graphs (figure 6.8 and 6.9). This symmetry is due to the shape of the piece. Because the piece has been created from a profile, which has been extruded. In consequence, the coordinates of the points of the profile are repeated but with different Z component.

The calculation of the velocities has been developed with a velocity of coating constant of 100 mm/s. It is possible to change this velocity of coating and calculate again the velocities of the robot and the external axis; configuring the parameter in the program.

In the graphs, it is possible to realize, that in the extreme points of the meander the velocity of the robot is equal to velocity of coating; meanwhile in the velocity of the external axis is zero. This is due to the fact that in the extreme points the movement of the robot is along Z axis, and the piece has been created from a profile extruded. In consequence if the robot is moved along Z axis, the component is kept in the same position on the external axis, and the perpendicularity of the spraying torch to the piece is maintained.

Finally, all the data calculated in Matlab are introduced to the simulation software (Roboguide). In Roboguide the translation to the system of reference is done through a function, which defines a frame where it is possible to place the new system of reference. In figure 6.10 a picture with the robot and the piece in the external axis, both developed in the simulation software, is showed.

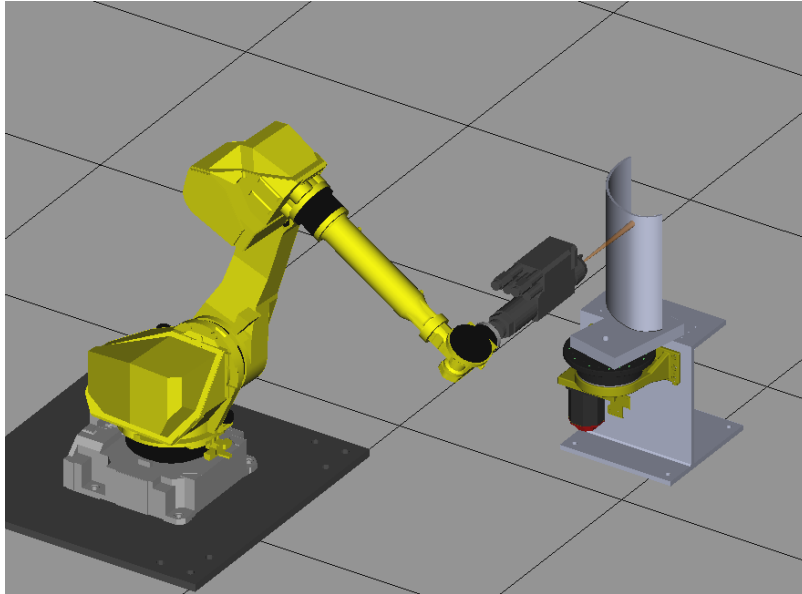


Fig. 6.10: Screen of Roboguide with blade formed by two arcs and external axis.

EXAMPLE 2 – Blade formed by two arcs with rotation center in position 2

In this second example, the same piece from the example one is going to be used. The difference will be that the rotation center is in other position (200, 0). The rotation center coincides with one of the center of the arcs, which forms the blade. In figure 6.11, the CAD piece and the piece in matlab with the normal vectors are showed; it has a different holder than the example 1.

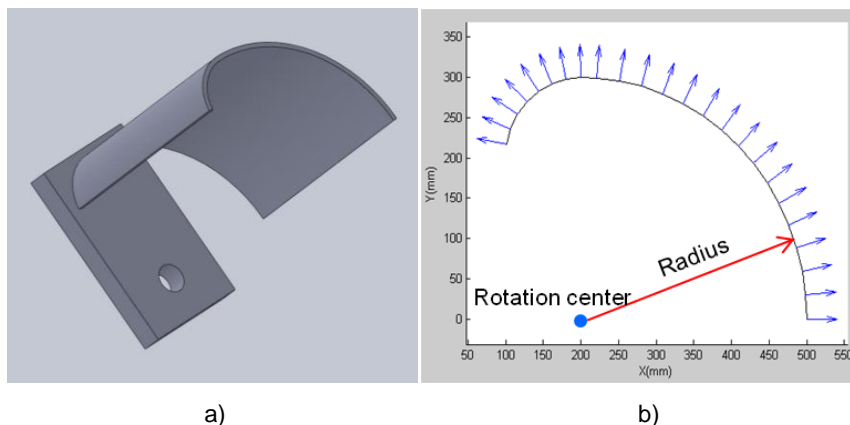


Fig. 6.11: Blade formed by two arcs in CAD format (a) and in Matlab with the normal vectors (b).

The normal associated vectors and the different positions of the external axis in each moment will be the same than in example 1 (figures 6.6 and 6.7). This is because it is the same piece, but with different rotation center. But the trajectory, which is going to follow the robot, will be completely different (figure 6.12).

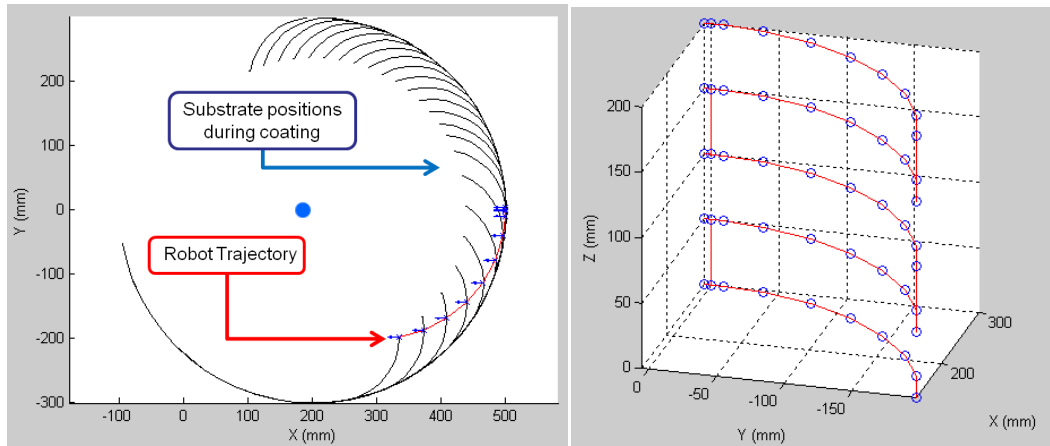


Fig. 6.12: Graphs with the trajectory of the robot.

The calculation of the velocities was performed for a coating velocity of 100 mm/s. The velocities of the robot and the external axis are showed in the figure 6.13.

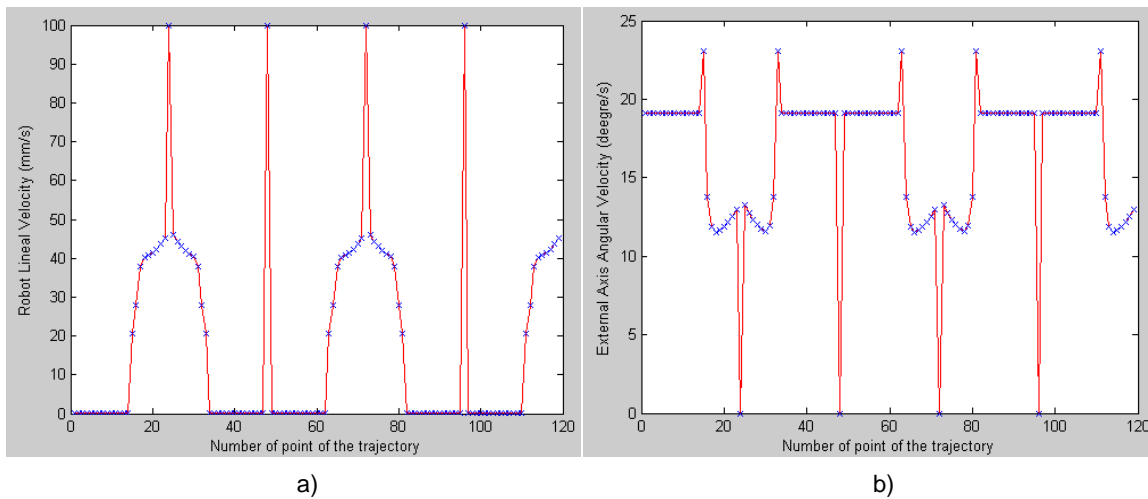


Fig. 6.13: Graphs with the velocities of the robot (a) and the external axis (b).

At the beginning of the trajectory the robot has a null velocity, while the external axis has a constant velocity. It is because the rotation center is positioned in one of the center of the arcs, which forms the blade. If a cylinder is rotating about its axis of symmetry, the normal associated vectors will be in the same direction of the torch in the same point of the space. In consequence, this is what happens in the first part of the trajectory of the robot; the arc of circumference, which forms the blade, is rotating about its axis of symmetry.

Finally, the data, which have been obtained from matlab, are introduced in the simulation software (Roboguide). In figure 6.10, the virtual cell with the piece can be seen.

With those two first examples, it is possible to realize of the importance of the rotation center. Depending on the position of the rotation center, the trajectory of the robot and the velocities will be completely different. In figure 6.14, graphs with the comparison between the velocities in both examples are showed. The red one shows the velocities with the rotation center in (200, 400), while the blue one shows the velocities with the rotation center in (200, 0).

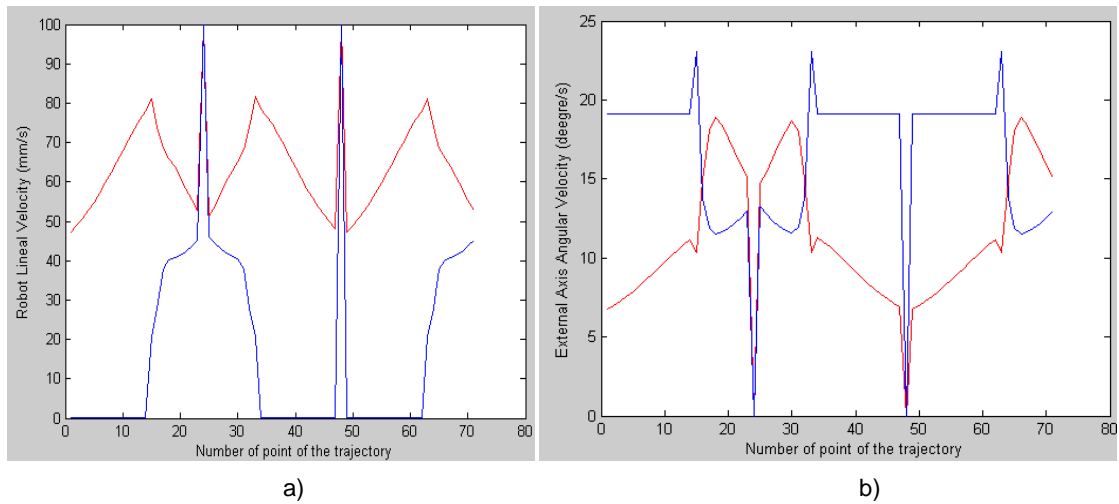


Fig. 6.14: Graphs with the velocities of the robot (a) and the external axis (b).

EXAMPLE 3 – Airfoil section NACA 5020

In this example, an airfoil section NACA 5020 is going to be coated completely. The profile of the airfoil is defined by a series of points, with this points the profile is going to be designed and extruded. The result of this process can be seen in figure 6.15.



Fig. 6.15: Airfoil section NACA 5020 in CAD format.

With the points, which define the profile, the associated normal vectors in matlab can be calculated. In this case the rotation center is set in the position (0, 0). The result is showed in figure 6.16.

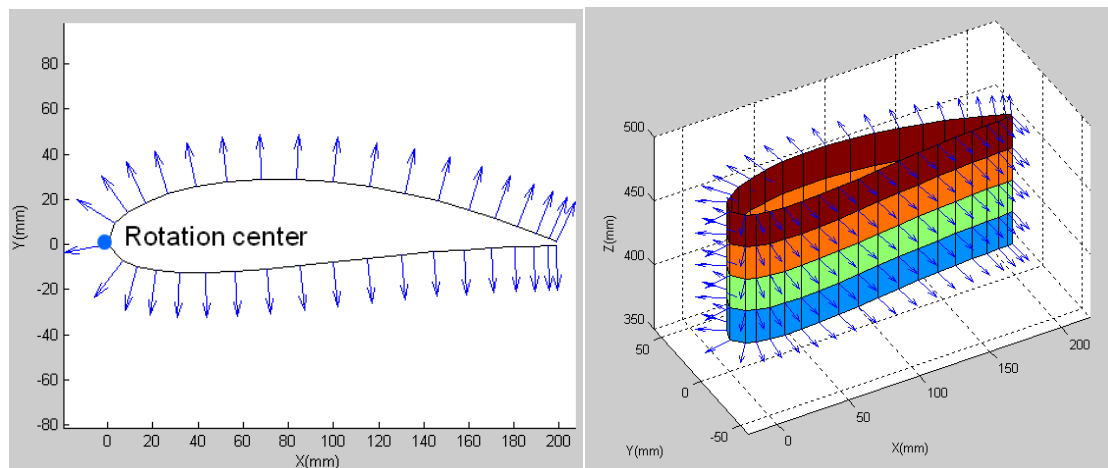


Fig. 6.16: Airfoil section NACA 5020 with its normal associated vectors.

The next step is to calculate the different positions of the external axis for each moment. In the figure 6.17, the graph has symmetry as in previous examples; it is because the piece is symmetrical along z axis.

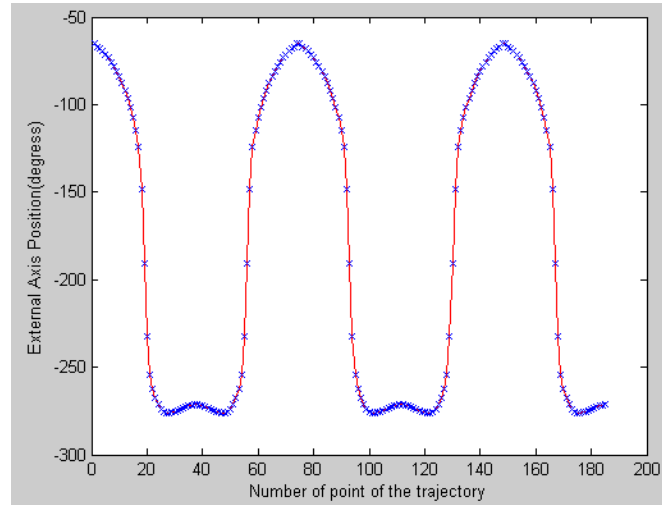


Fig. 6.17: Graph with the angular position in each instant of the external axis.

With the different positions of the external axis and the normal associated vectors, the trajectory of the robot can be calculated. In figure 6.18, the trajectory of the robot is showed.

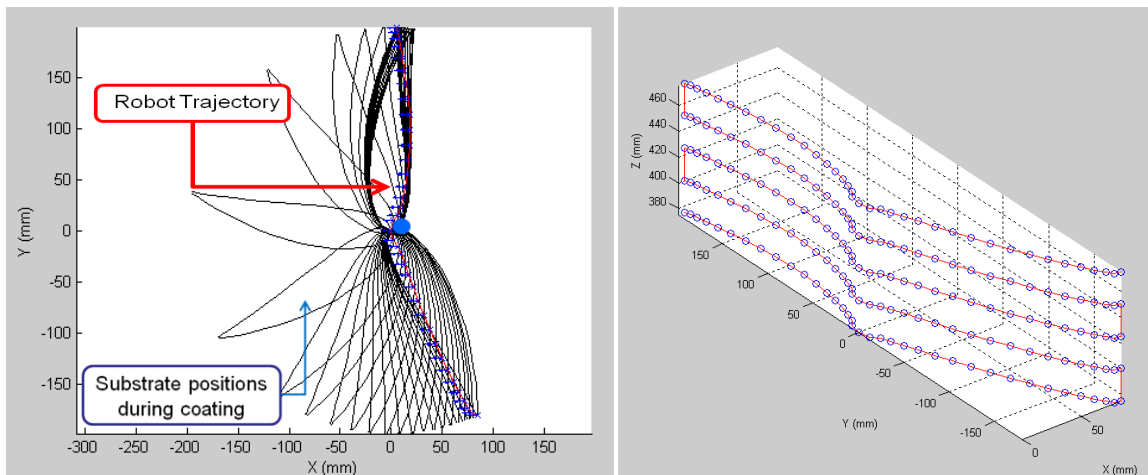


Fig. 6.18: Graphs with the trajectory of the robot for an airfoil section.

The velocities of the robot and the external axis have been calculated for a coating velocity of 100 mm/s (figure 6.19). The graph with the angular velocities has high values around 400°/s. The external axis, which is installed in the IFKB, has a limitation speed of 160°/s. In consequence, a solution has to be found to rotate the piece faster or get slower the velocity in the external axis during the coating process.

Finally, as in the other examples, the data, which have been calculated in matlab, are introduced in Roboguide. In this example, two simulations are going to be made, one with the velocity limitations of the external axis and another without limitations. In the figure 6.20, the virtual cell with the piece in the external axis can be seen.

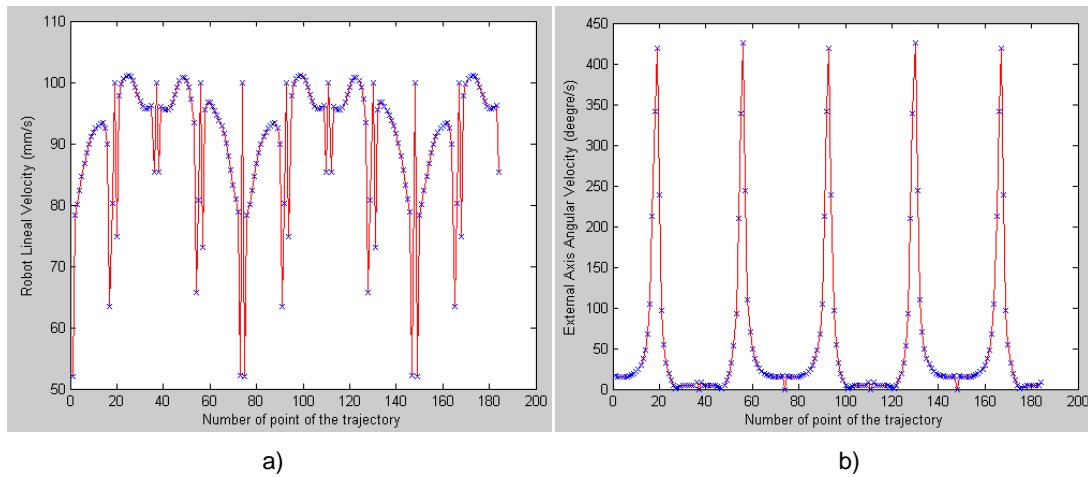


Fig. 6.19: Graphs with the velocities of the robot (a) and the external axis (b).

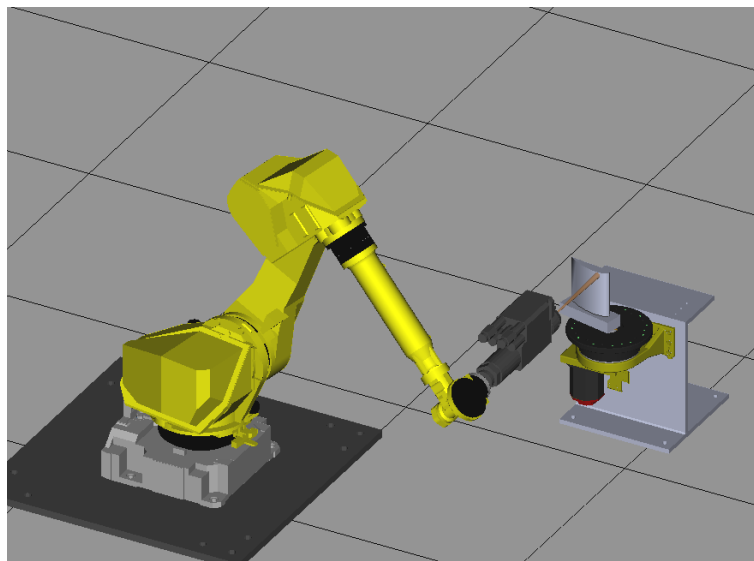


Fig. 6.20: Screen of Roboguide with airfoil section and external axis.

EXAMPLE 4 – Sigma for injection molding application (Knetter)

The sigma blade (Knetter) is defined by a cloud of points (figure 6.21 b), with this cloud of point, the representation of the piece in CAD format can be obtained (figure 6.21 a). This piece is not extruded; it is not symmetrical along z axis.

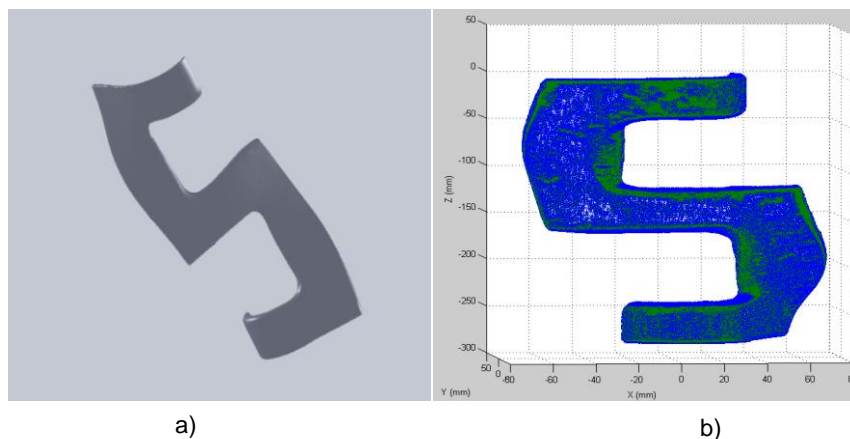


Fig. 6.21: Knetter in CAD format (a) and with a cloud of points (b).

To obtain the points of the trajectory on the piece, a program has been developed. The program uses the nearest neighbor algorithm (see appendix 10.3) to classify the points and its normal associated vectors on the lines of the meander trajectory. Later, the interpolation is used to obtain the final trajectory. In figure 6.22, the different steps of the process of classification and interpolation are showed.

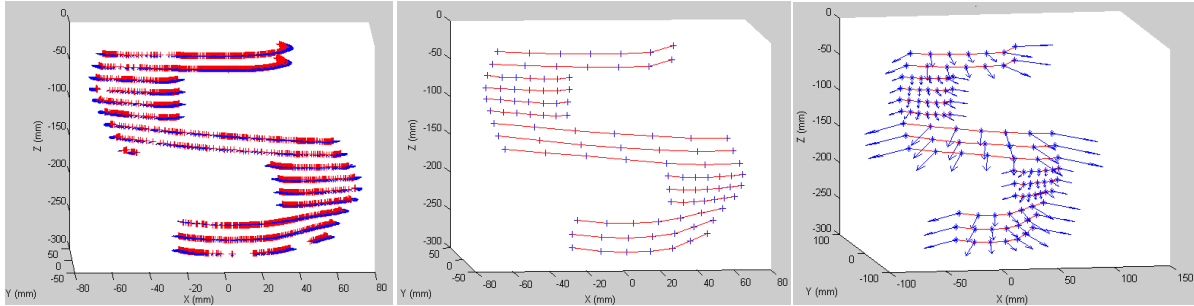


Fig. 6.22: Steps in the process to obtain the trajectory on the piece and the normal associated vectors.

With the points of the trajectory and its normal associated vectors, the different positions of the external axis can be calculated, to align the vector with the direction of the torch (figure 6.23).

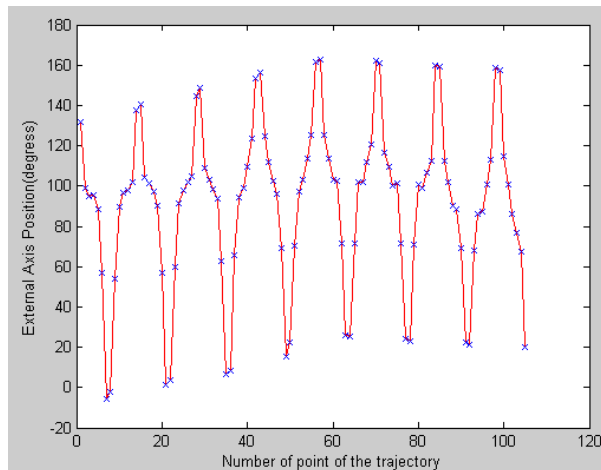


Fig. 6.23: Graph with the angular position in each instant of the external axis.

As told at the beginning, the graph in the figure 6.23 is not symmetric as in other cases. This is because the piece is not symmetrical respect Z axis. The next step is to calculate the trajectory of the robot (figure 6.24).

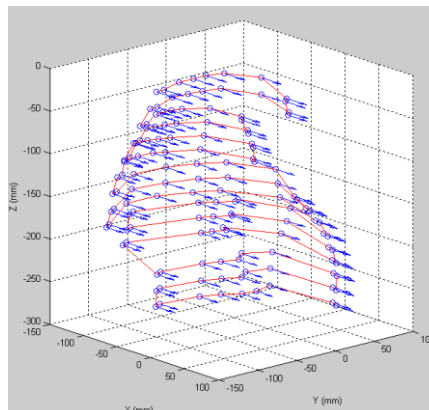


Fig. 6.24: Trajectory of the robot for a Knetter.

The rotation of the piece has been done in the position (-50, -50), while the calculation of the velocities of the robot and the external axis has set for a coating velocity of 100 mm/s. The graphs with the velocities can be seen in figure 6.25. As in the rest of the cases, the data calculated in matlab are introduced in roboguide to simulate the coating process.

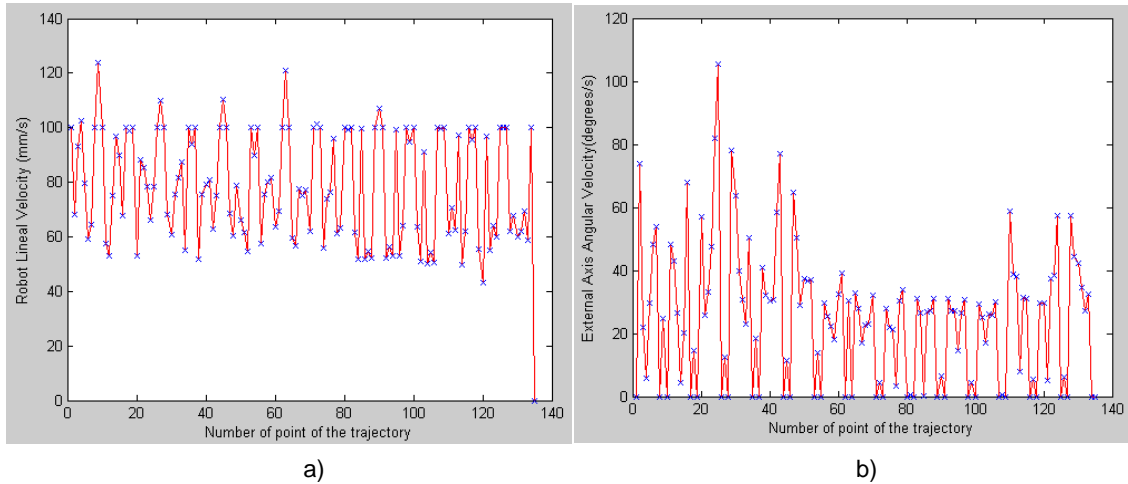


Fig. 6.25: Graphs with the velocities of the robot (a) and the external axis (b).

EXAMPLE 5 – Blade for printing industry (KBA)

This is the last example for this approach. The last piece is a blade for printing industry, which has been designed in a CAD format. It can be seen in figure 6.26.

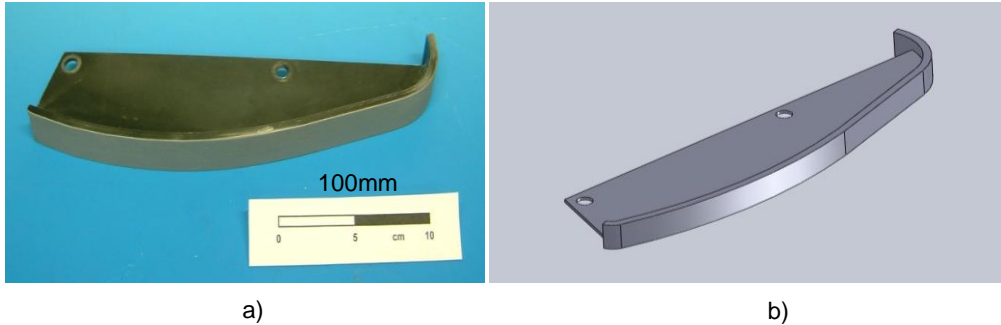


Fig. 6.26: Blade for printing industry, the real one (a) and a CAD format (b).

With the CAD piece, the points which define the profile of the component can be taken. Introducing these points in matlab, the normal associated vectors can be calculated (figure 6.27).

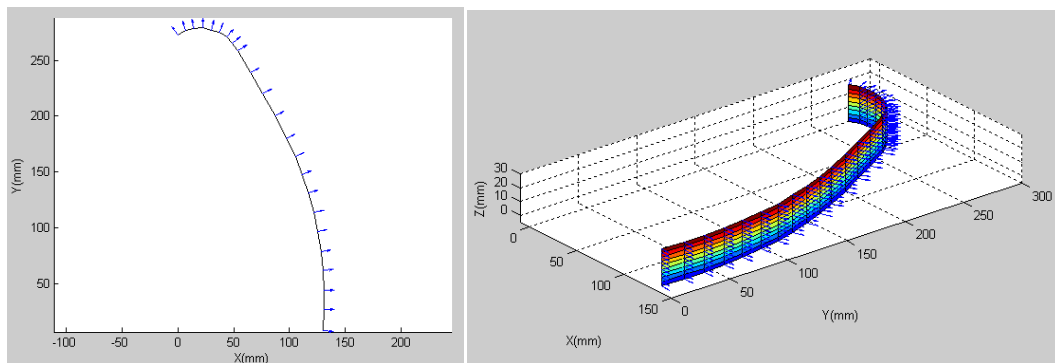


Fig. 6.27: Blade for printing industry with its normal associated vectors.

With the normal vectors, the different positions of the external axis can be calculated (figure 6.28) for a rotation center in the position (0, 300). In this case, the blade for printing industry is an extruded piece (symmetrical along Z axis). In consequence, the graph of the position of the external axis is symmetrical as the velocities and the trajectory of the robot.

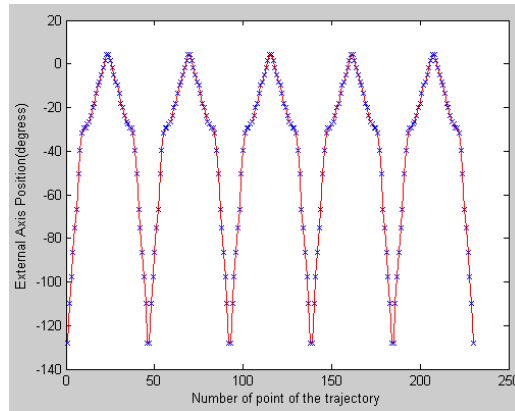


Fig. 6.28. Graph with the different position of the external axis for the KBA.

As in the previous cases, the trajectory of the robot (figure 6.29) can be calculated with the position of the external axis. Later, the velocities of the robot and the external axis (figure 6.30) can be calculated with the previous data.

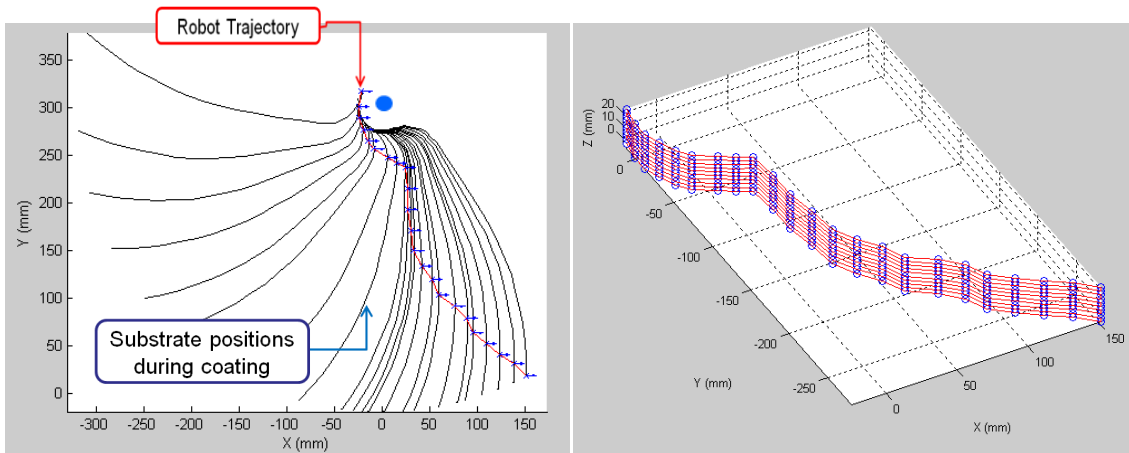


Fig. 6.29: Trajectory of the robot for a KBA piece.

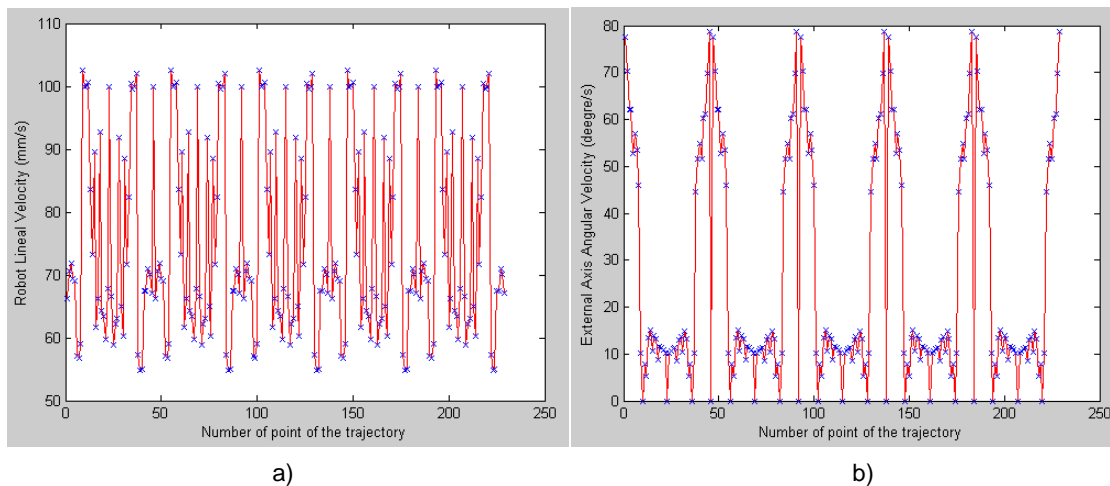


Fig. 6.30: Graphs with the velocities of the robot (a) and the external axis (b).

The calculation of the velocities has been done for a coating velocity of 100 mm/s. Finally, as in the previous examples the data have been introduced in roboguide to make the simulation of the coating process (figure 6.31).

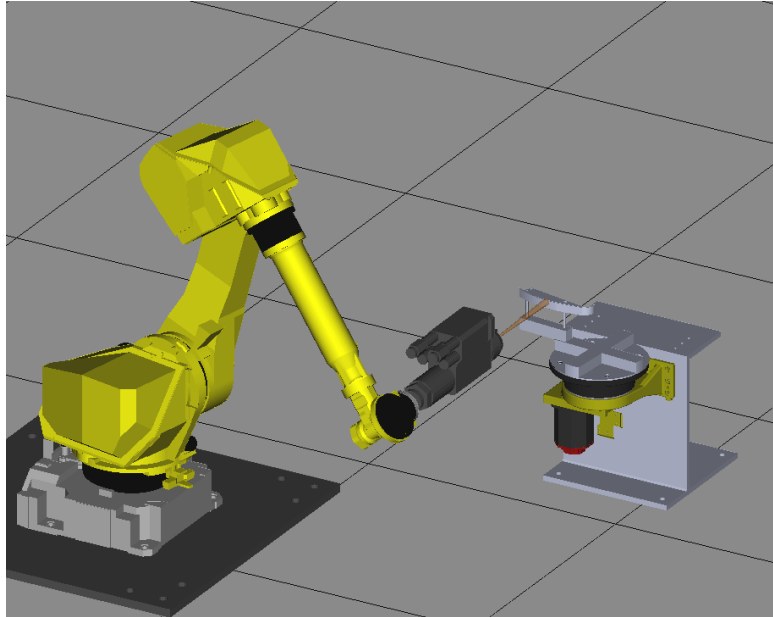


Fig. 6.31: Screen of Roboguide with blade for printing industry and external axis.

6.2 Approach II

6.2.1 Concept

In the first approach, the movement of the torch was limited to the X and Y axes. In this second approach, one degree of freedom is going to be added to the system. In consequence, the torch will rotate and orientate with the angle yaw (figure 6.32).

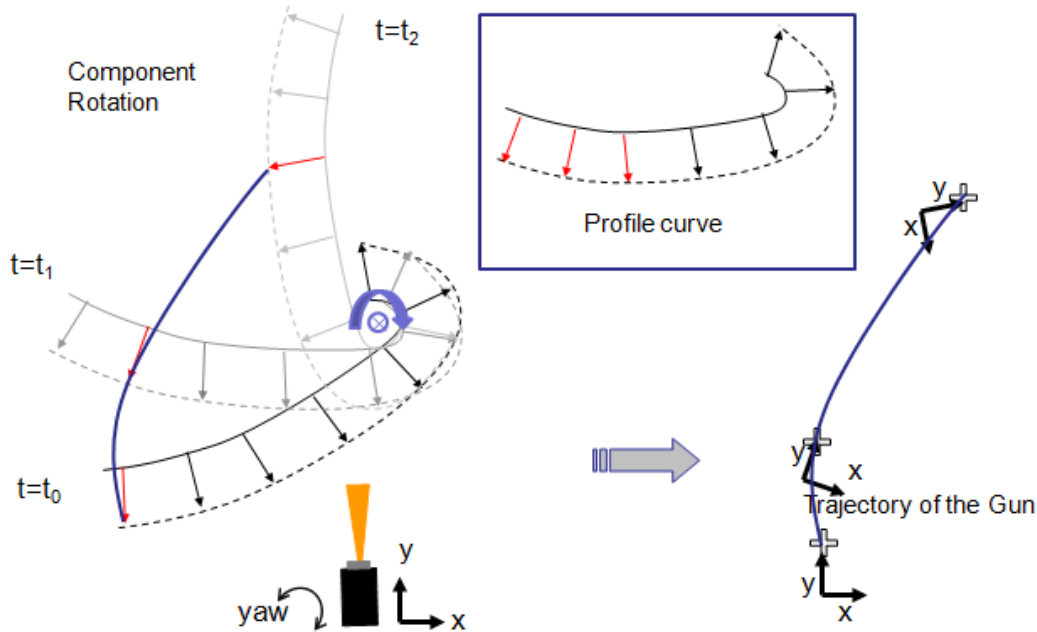


Fig. 6.32: Scheme of the approach II.

In this case, the angle of rotation of the piece in each moment can be chosen, because it is not necessary that the normal associated vector has the same direction than the torch. The torch can rotate and orientate in the same direction than the vector of the piece. In figure 6.32, different position of the piece and the trajectory of the robot can be seen.

This scheme shows how to coordinate the movement of the robot and the external axis. But it is necessary to calculate the coating velocity. In this approach, the coating velocity is more difficult to calculate than in the first approach. Because the relative velocity between the robot and the external axis is more complex, in this case the robot has a linear velocity (movement in the plane XY) plus angular velocity (orientation in the angle yaw). The calculation of the velocities for the robot and the external axis can be proposed as further developments of this project.

6.2.2 Steps to design the trajectory

The steps to design the trajectory are going to be the same as in the first approach, the velocities, due to the complexity of the coordinated movement, were not calculated. Removing the step of the calculation of the velocities in the figure 6.4, the scheme with the steps to design the trajectory will be ready (figure 6.33).

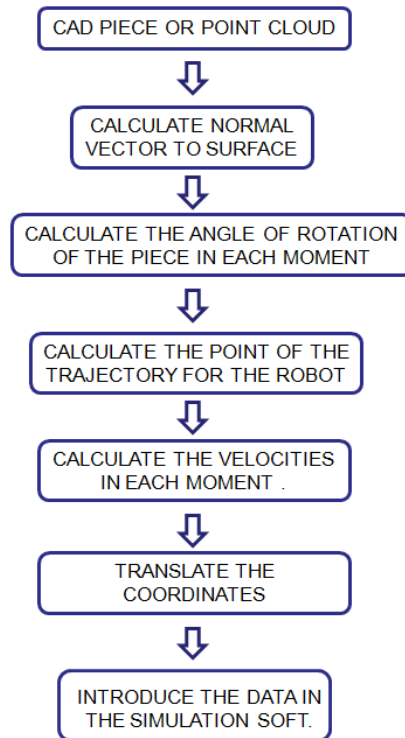


Fig. 6.33: Scheme with the steps to design the trajectory for the approach II.

The calculation of the different points of the trajectory is going to be made in matlab, while the simulation of the coating process is going to be made in roboguide.

6.2.3 Example

For this approach, just one example has been designed. The example is going to be a blade formed by two arcs of circumference. This piece was used in the first approach with two different rotation centers. In figure 6.5, the piece in CAD format is showed.

The rotation center is going to be positioned in (200, 400). The normal associated vectors to the piece are going to be the same as in example 1, so in figure 6.6, the piece with the normal vectors is showed.

In this approach, the different position of the external axis can be selected by the programmer. Because a degree of freedom has been added to the system. An increase of 10° each time has been selected to rotate the piece. The different position of the piece can be seen in figure 6.34. Finally, with the normal vectors and the different position of the external axis, the trajectory of the robot can be calculated (figure 6.35).

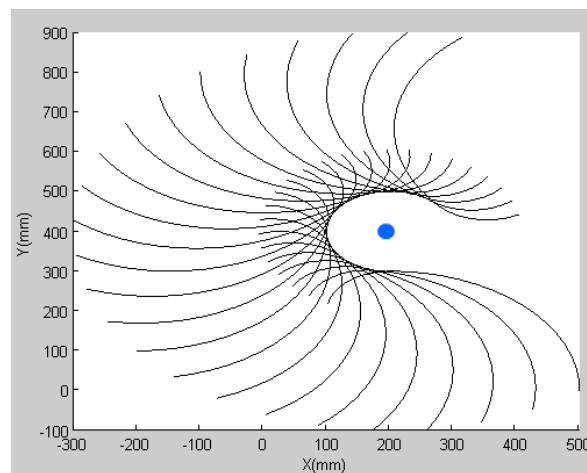


Fig. 6.34: Positions of the substrate during the coating process.

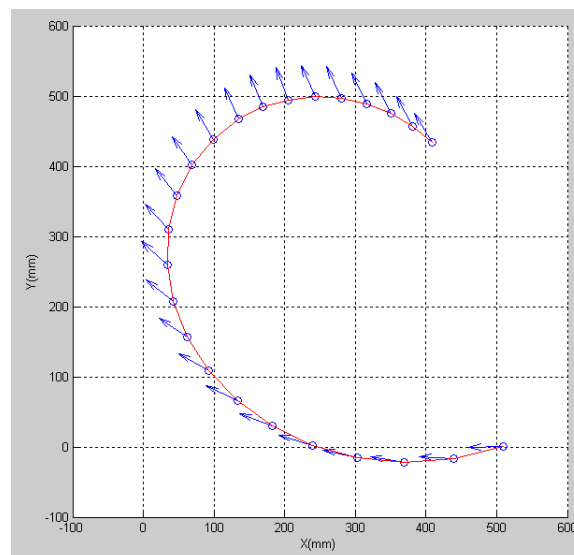


Fig. 6.35: Trajectory of the robot for the approach II.



6.3 Discussion

In this chapter, two ways to coordinate the movement of the robot and the external axis have been seen. Also, some examples for both approaches have been seen. In this section, a short conclusion will be exposed.

The rotation center is an important parameters to calculate the trajectory of the robot and the velocities. This importance has been seen in the previous examples, although especially in the first, second and third examples of the first approach. In the two first examples, the change of the velocities and the trajectory with the rotation center has been tested, while in the third example the limitation of the velocities depending on the position of the point, has been seen.

In the example two, the advantages of placing the rotation center in one of the arcs, which forms the piece, have been seen. The main advantages are: simplifying the movement of the robot and keeping the velocity constant in one part without many variations. In consequence, one idea could be to approximate the profile of the piece with arcs of circumference and linear segments. Thus, the election of where to place the rotation center will be easier.

In the second approach, to add a degree of freedom to the system makes that the movement of the robot will be more complex. When the movement of the robot is complex it is more probable to reach singularity positions, which restrain the movement. In consequence, it would be desirable to use the first approach, doing previously a study about where to place the rotation center.



7. Conclusions

The importance of industrial robots nowadays is due to: costs reduction, quality increasing, etc. The use of industrial robots in thermal spraying operations allows better control of heat and mass transfer to the substrate and precision during the spraying operation, and therefore the achievement of quality requirements.

The design of trajectories for robots used for thermal spraying operations can be performed using various methods: on-line, off-line or hybrid programming. Commercial simulation softwares (off-line programming) are very useful to support robot programming, but they normally need filtration and modifications for the automatically generated trajectories in order to obtain optimized trajectories for each component geometry. In chapter five a list of criteria to evaluate simulation softwares has been developed. Using these evaluation criteria, some of the most important simulation softwares available on the market have been analyzed.

Programs to calculate trajectories for thermal spraying and lacquered lacquered operations have been developed in Matlab. The points of the trajectory can be verified through a graph. Simulation softwares allow testing the trajectories in a virtual environment without using the real robot. This allows to check if the translation to the robot's language from Matlab has been successful.

When the piece has a complex geometry, the coating process cannot be performed in one operation, but it is necessary to change the component position and in tempt the coating in order to reach the whole area of the substrate. To avoid this, the implementation of an external axis has been studied. Matlab and Roboguide have been used to perform the calculations necessary to obtain the trajectory to coordinate the robot with the external axis.

Two approaches have been developed to coordinate the movement between the robot and the external axis. If the robot is given three degrees of freedom, like in the second approach presented in this work; lineal movement X and Y and rotation around its Z axis, the movement of the robot is complicated and can lead to singularities and unreachable positions. On the other hand, as presented in the first approach, if the robot moves linearly in the XY plane and the external axis performs the rotation, the robot movement is highly simplified.

The importance of the location of the rotation center respect to the piece has been seen in the different examples of the chapter 6. Optimizing the position of the rotation center improves the trajectory of the robot and the velocities of the robot and the external axis. In the same way, by simplifying the component geometries in circumferences arcs and lines, the coordinated movement can be optimized.



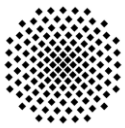
8. Further Developments

At the end of this project, different ways to design trajectories and the implementation of an external axis have been development. In the chapter 4 (design of trajectories for planar substrates), some trajectories have been designed and tested in the robot, but not in a real thermal or lacquer spraying processes. One possible further development could be, to check the trajectories designed.

In the chapter 6 (implementation of an external axis) the rotation center influences in the velocity and robot trajectory during the coating process. A further development would be to optimize the rotation center to obtain the best robot trajectory and velocity during the coating process.

In the implementation of an external axis, the angular velocity of the external axis is limited to $160^\circ/\text{s}$. Another further development would consist on a solution finding to increase the angular velocity. One possible solution, a gear, which multiplies the velocity, could be installed.

Finally, the implementation of an external axis has been developed just in the computer with simulations. The next step would be to realize the trajectories in a real robot with an external axis and check how the substrate has been influenced after the coating process.



9. Bibliography

- ACEDO05 Acedo Fornell M.C., "Thermal sprayed coatings on piston rings, processing and characterization", Augsburg 2005
- ANGELES03 Angeles J., "Fundamentals of robotic mechanical systems: theory, methods and algorithms", Second edition. Springer 2003.
- BACH06 Bach F.W., Laarman A., Wenz T., "Modern surface technology", WILEY-VCH 2006, ISBN: 3-527-31532-2
- BARRIENTOS07 A. Barrientos, L.F. Peñin, C. Balaguer, R. Aracil, "Fundamentos de Robotica, 2ed.", McGraw-Hill, ISBN: 8448156366, 2007.
- CANDEL05 Candel A., Cordoba F., "Development of simulation tools and automatic generation of 3D trajectories for thermal spraying applications", University of Stuttgart 2005.
- CANDEL06 Candel A., Gadow R., "Optimized multiaxis robot kinematic for HVOF spray coating on complex shaped substrates", Elsevier B.V. 2006.
- CANDELDIP Candel, Antonio. "Optimized robot kinematics for thermal spray and lacquer coatings", Universität Stuttgart, Institut für Fertigungstechnologie keramischer Bauteile, Diploma Thesis, Stuttgart.
- CRAIG90 Craig J.J., "Introduction to robotics, mechanism and control", Second edition. Addison-Wesley publication Co. New York, 1990.
- DAVIS04 J.R. Davis and Associates, "Handbook of Thermal Spraying Technology". ASM International 2004. ISBN 0-87170-795-0.
- DAVISEVA L. Davis and G. Williams, "Evaluation and selecting simulation software using the analytic hierarchy". Integrated Manufacturing Systems 5(1):23-32.
- DENAVIT64 Denavit J. and Hartenberg R.S., "Kinematic synthesis of linkage, mechanical engineering", New York 1964, Mc-Graw Hill.
- DUDA01 Duda R.O., Hart P.E., Stork D.G., "Patter classification (2° Edition)", Ed. John Wiley and Sons 2001.
- DYKHUIZEN98 Dykhuizen R.C. and Smith M.F., "Gas dynamic principles of cold spray", ASM International 1998
- FAUCHAIS04 Fauchais P., Understanding Plasma Spraying. J. Phys. D: Appl. Phys. 37 R86-108 (2004).
- FLORISTAN07 Floristan M., "Optimization of the plasma spray process for oxide ceramic coatings used on printing rolls", Institute for manufacturing technologies of ceramic components and composites (IMTCCC), University of Stuttgart 2007



- FRUTOS09 Alejandro Frutos, "Numerical analysis of the temperature distribution and Offline programming of industrial robot for thermal spraying", Universität Stuttgart, Institut für Fertigungstechnologie keramischer Bauteile, Final Thesis, Stuttgart 2009
- HANDLINGMAN "Fanuc robot series R-30iA, Handling Tool, Operators manual" Fanuc Robots America Inc. 2007.
- HOLDER90 K. Holder, "Selecting Simulation Software". OR Insight, Vol.3 No.4, pp. 19-24 (1990).
- ISO8373 EN ISO 8373:1996 Manipulating Industrial Robots – Vocabulary
- KERAMAS99 Keramas J.G., "Robot technology fundamentals", Demar publisher, San Francisco 1999; ISBN: 0-8273-8236-7.
- MANKAREL "Fanuc Robotics System, R-30iA Karel reference manual", version 7.30, Fanuc Robotics America inc. 2007.
- MANPX08 PaintiXen version 6.2.0 – User Manual. 2008.
- MANSRS08 Stäubli Robotics Studio – User Manual. 2008.
- MANV+ "V+ Language user's guide", version 12.1, Adept Technology, inc.
- MANVPAINTE "V_Paint Graphic", version 2.1.A, Stäubli 2001
- MANVTRAJ "V_Trajsig", Stäubli 1997
- MCKERRON91 McKerron P.J., "Introduction to Robotics", Addison Wesley, 1991.
- MITSI04 Mitsi S., Bouzakis K.D. and Mausour G., "Off-line programming of an industrial robot for manufacturing", Springer-Verlag London limited 2004.
- NIKOUKARAN98 J. Nikoukaran, V. Hlupic, R. J. Paul, "Criteria for Simulation software Evaluation". Dept. of Information Systems and Computing, Brunel University, UK. 1998.
- PAUL07 Paul D. A. Jones, Stephen R. Duncan, Tim Rayment and Patrick S. Grant, "Optimal robot path for minimizing thermal variations in a spray deposition process.", IEEE Transactions on control systems technology, vol. 15, N° 1, January 2007.
- PAWLOWSKI95 Pawlowski L., "The science and engineering of thermal spray coating", John Wiley and Sons, 1995
- QUICK_KUKA "Quick guide – Kuka.sim", Kuka company
- RIA Robotics Industries Association
- ROBDAT09 Roboguide – Datasheet 2009. ROBOGUIDE(E)-04, 2009.11
- SCHNEIDER06 K.E. Schneider, V. Belashchenko, M. Dratwinski, S. Siegmann, A. Zagorski, "Thermal Spraying for Power Generation Components". WILLEY-VCH 2006. ISBN 3-527-31337-0.



- SICILIANO09 B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, "Robotics. Modelling, Planning and Control", Springer, ISBN 978-1-84628-641-4, 2009
- VUORISTO Vuoristo P., Tuomin J., Numinen J., "Laser coatings and thermal spraying – process basics and coatings properties"
- WEBASM <http://www.asm-int.org/tss/glossary/t.htm>
- WEBEASY <http://www.easy-rob.com>
- WEBFANUCEU <http://www.fanucrobotics.com/>
- WEBFANUCJP <http://www.fanuc.co.jp/en/profile/headquarters/index.html>
- WEBFANUCSOF <http://www.fanucrobotics.com/products/vision-software/AtoZ.aspx>
- WEBKUKACOMP <http://www.kuka-robotics.com/en/company/>
- WEBKUKASOF <http://www.kuka-robotics.com/en/products/software/>
- WEBSTÄUBLI <http://www.staubli.com/en/robotics/>
- WEBSULZER http://www.sulzermetco.com/desktopdefault.aspx/tabid-1740/3392_read-5304/



10. Annex

10.1 Relative velocity between two particles

Consider two particles, A and B, which move in space and are r_a and r_b its position vectors to the origin of a given referential frame (world frame). The velocities in A and B respect to this referential frame are:

$$\vec{V}_a = \frac{d\vec{r}_a}{dt} \quad \vec{V}_b = \frac{d\vec{r}_b}{dt}$$

The position vectors (relative) particle B with respect to A and A to B are defined by:

$$\vec{r}_{ab} = \vec{r}_a - \vec{r}_b \quad \vec{r}_{ba} = \vec{r}_b - \vec{r}_a$$

The velocities relative of particle B with respect to A and A with B are:

$$\vec{V}_{ab} = \frac{d\vec{r}_{ab}}{dt} \quad \vec{V}_{ba} = \frac{d\vec{r}_{ba}}{dt}$$

Since $\vec{r}_{ab} = -\vec{r}_{ba}$ it is also true that $\vec{V}_{ab} = -\vec{V}_{ba}$, so the relative velocity vectors of B with respect to A and A with respect to B are equal and opposite. Performing the derivatives:

$$\frac{d\vec{r}_{ab}}{dt} = \frac{d\vec{r}_a}{dt} - \frac{d\vec{r}_b}{dt} \quad \frac{d\vec{r}_{ba}}{dt} = \frac{d\vec{r}_b}{dt} - \frac{d\vec{r}_a}{dt}$$

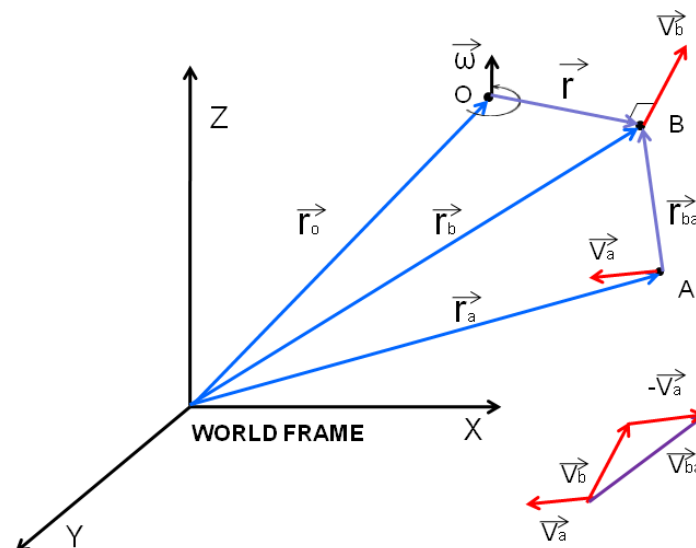


Fig. 10.1: Scheme for relative velocity between two particles.

$$\boxed{\vec{V}_{ab} = \vec{V}_a - \vec{V}_b \quad \vec{V}_{ba} = \vec{V}_b - \vec{V}_a}$$

10.2 Using auxiliary frames

The translation of the coordinates of one point in the space, from one system of reference to another system of reference is a complex task. It requires a complex math matrix calculation. Nowadays, the industrial robots are implemented with software, which makes this calculation easier.

The system defines the location and orientation of positional data relative to a user-defined frame of reference, called user frame (figure 10.2). Normally all the robot systems have three basic systems of references:

- **World frame:** the world frame is predefined for each robot. It is used as the default frame of reference. The location of world frame differs for each robot model.
- **Tool frame:** the programmer defines the position of the TCP (tool center point) relative to the faceplate of the robot.
- **User frame:** the programmer defines user frame. The location of the user frame represents distances along the x-axis, y-axis, and z-axis of the world coordinate system; the orientation represents rotations around those axes.

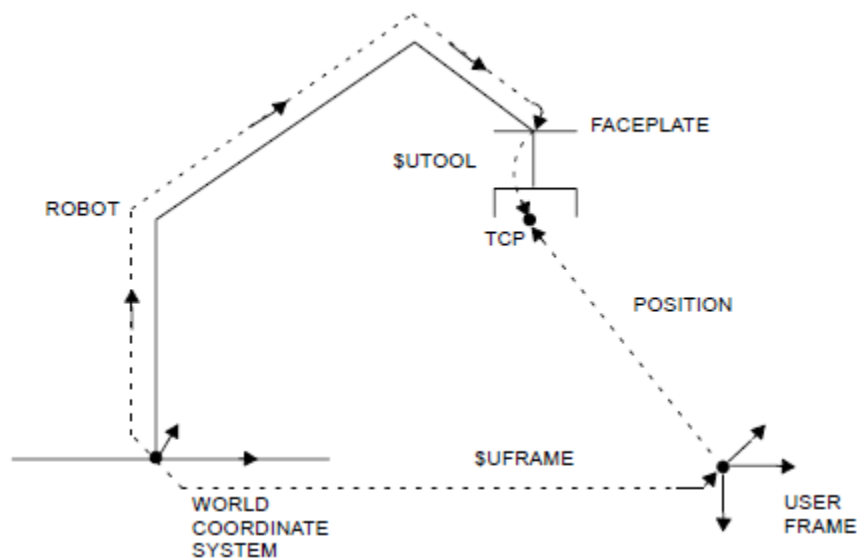


Fig. 10.2: Scheme about coordinates using auxiliary frames [MANKAREL].

Using kinematic equations, the controller computes its positional information based on the known world frame, user frame and tool frame.

For example, a program describes the trajectory of the robot to coat a piece and the points are referenced to one point of the piece. To translate the coordinates of the program to the robot is not necessary to change the coordinates; just defining a user frame, which specifies where is the piece respect the world coordinate system, is enough. The robot software will make the necessary calculation to obtain the coordinates of the points referenced on the world coordinate system.

10.3 Nearest neighbors algorithm

Nearest neighbor search (NNS), also known as proximity search, similarity search or closest point search is an optimization problem for finding closest points in metric spaces. At the beginning a series of data are classified in different classes and then a new point, which is not classified, is introduced. The NNS will classify this point in one of the classes defined at the beginning [DUDA01].

The parameter to classify the different points is going to be the distance. Depending on the distance between the point and the different classes the point will be classified in one of them. In figure 10.3, a scheme with three classes and a point, which is not classified, is showed.

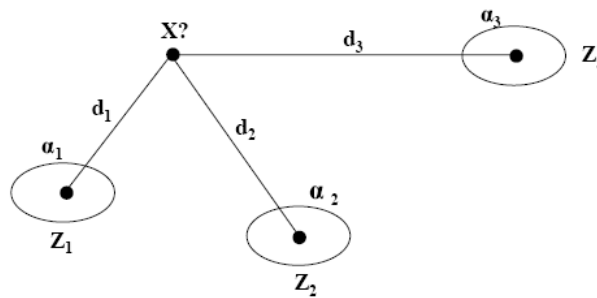


Fig. 10.3: Scheme applied the NNS with three classes.

The point X will be classified in the class, which will be nearer from it. With this method different points in the space can be classified in different classes predefined by the programmer following this method.



10.4 Programs for planar trajectories

In this section, the different programs, which have been developed to coat planar substrates, are mentioned. The basic scheme of the programs in matlab is showed in figure 10.4.

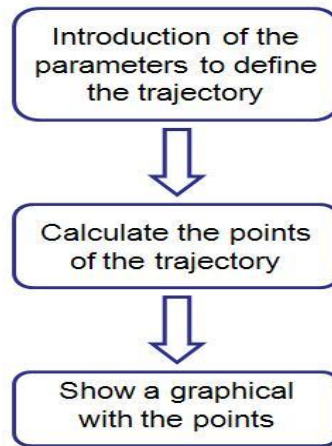


Fig. 10.4: Scheme of the matlab program for planar trajectories.

The name of the programs in matlab with the calculation of the trajectories, which have been showed in chapter 4, are:

- meander_HA_v11.m: horizontal meander type A.
- meander_VA_v11.m: vertical meander type A.
- meander_HB_v11.m: horizontal meander type B.
- meander_VB_v11.m: vertical meander type B.
- meander_HC_v11.m: horizontal meander type C.
- meander_VC_v11.m: vertical meander type C.
- meander_triangular_v13.m: triangular trajectory.
- meander_HV_v10.m: horizontal variable meander.

The program in matlab have been translated to the robot's language, in this case the robot's language is the V+. The name of the programs in V+ are:

- M_HYA.v2: horizontal meander type A.
- M_VYA.v2: vertical meander type A.
- M_HYB.v2: horizontal meander type B.
- M_VYB.v2: vertical meander type B.
- M_HYC.v2: horizontal meander type C.
- M_VYC.v2: vertical meander type C.
- M_HVar.v2: horizontal variable meander.

10.5 Programs for the external axis

In this section, the programs developed for the coordination movement between the robot and the external axis are explained. Different programs have been developed for both approaches. For the first approach, the pieces, which have symmetry along Z axis (extrude), are going to be analyzed. These pieces are:

- Blade formed by two arcs of circumference.
- Airfoil section NACA 5020.
- Blade for printing industry (KBA).

The Kneiter is going to be analyzed separately. This is because, this piece is not symmetrical along Z axis, and the programs developed to obtain the trajectory and the velocities are different from the other pieces, which are extruded.

For each piece a function in matlab has been developed a, which has the profile of the piece and extrudes this profile. The names of the function are:

- p_2arc.m: profile of the blade formed by two arcs of circumference.
- p_helice2.m: profile of the airfoil section NACA 5020.
- p_last.m: profile for blade for printing industry (KBA).

The result is a matrix of points of the piece, which the robot has to follow lately. This matrix is introduced in a general program, which calculates the trajectory and the velocities of the robot and the external axis. Depending on the piece, which wants to be introduced, is necessary to modify the name of the function in the general program.

- rot_gun_xy.m: general program to calculate the trajectory of the robot and the velocities.

In this program, the rotation center can be modified and other parameters as offset of the meander, etc.

The program, which makes the calculation of the Kneiter, is completely different to the other programs. This is because the kneter doesn't have symmetry along Z axis, so the profile cannot be introduced and extruded. The cloud of points, which defines the kneter, is going to be introduced in a file with format .STL (CAD format). A function in matlab, which translates the file in .stl to a matrix of points with its normal associated vector in each point (figure 10.5), has been developed. The name of the function is "stlread.m".

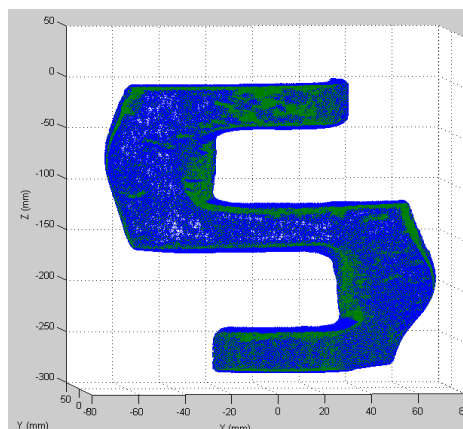


Fig. 10.5: Matrix of points with its normal associated vector in each point, obtained using "stlread.m".

The next step is to filter the points of the piece. The nearest neighbor algorithm is going to be used for filtering the points of the piece. The classification consists on defining the meander and point by point look to what line of the meander below each point. If the point doesn't belong to any line then it will be eliminated. A margin of error will be defined for one parameter to catch points, which are nearer of the line of the meander but not exactly in this line. For this purpose a function called "clasificar.m" has been designed. In figure 10.6, a scheme with the basic operation of this algorithm is showed.

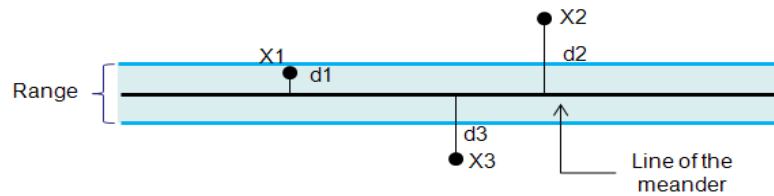


Fig. 10.6: Scheme of the application of the nearest neighbor algorithm.

A range (margin of error) is defined in this algorithm. If the points are inside of this range then the points will be selected for this line of the meander. In figure 10.6, three points are classified but just the point X1 has the distance inside of the range available. In consequence, the point X1 will be selected for the meander; meanwhile the points X2 and X3 will be eliminated. Applying this algorithm to all the matrix of points, the obtained results is showed in figure 10.7.

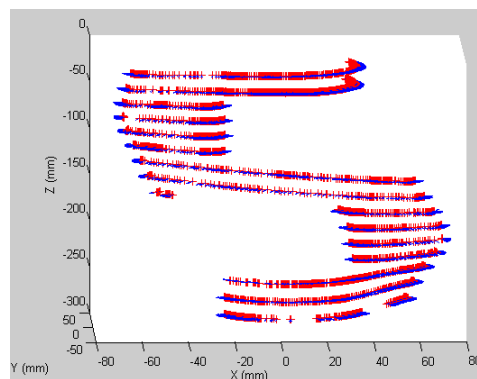


Fig.10.7: Matrix of points after using the function "clasificar.m".

Once the points, which belong to the lines of the meander, have been obtained. The next step is interpolated each line of the meander with the clouds of points associated to each one. With this interpolation a meander with the shape of the can be obtained. These lines obtained from the interpolation are going to be defined by number of points, which can be set by the programmer. The function, which makes this calculation, is called "interpolation.m". In figure 10.8, the results to apply this function to the previous matrix of points, is showed.

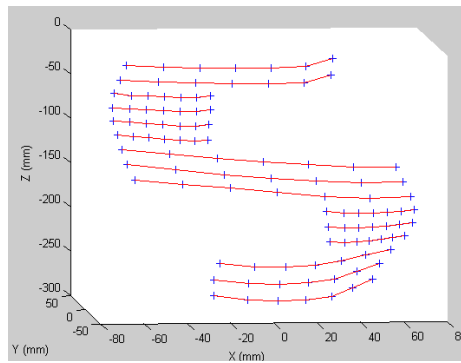


Fig. 10.8: Matrix of points obtained after using the function “interpolation.m”.

The next step is to assign to the points obtained with the previous function the normal associated vectors. Taking the matrix obtained with the function “clasificar.m” and applying the nearest neighbor algorithm, it is possible to associate to each point the normal associated vector. The function, which develops this classification, is called “normvec.m”. In figure 10.9, the result to apply this function is showed.

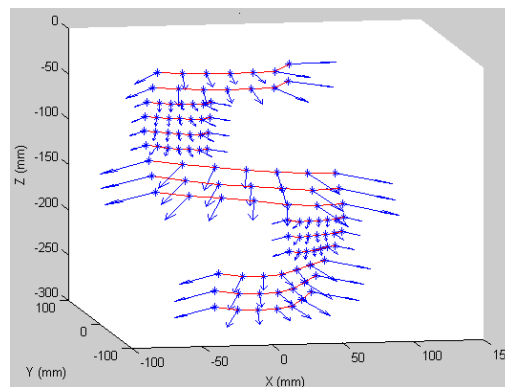


Fig. 10.9: Matrix of points and normal associated vectors after using the function “normvec.m”.

Finally, the matrix with the points and normal vectors of the trajectory on the piece is obtained. From here, the process is similar to the previous programs for the other pieces. Two more functions have been designed to obtain the trajectory of the robot (“calctra.m”) and the velocities of the robot and the external axis (“calcvel.m”). The program “knetest.m” is going to execute all this function step by step, obtained the results showed in chapter 6.

For the second approach, just one example has been developed. The example is the blade piece formed by two arcs of circumference. Using the function developed previously to obtain the matrix of points (“p_2arc.m”), the next step is introduced this function in a general program. The general program will make the calculation to obtain the trajectory of the robot and the velocities. The program is called “rot_gun_xyw.m”. In this program is possible to configure different parameters like the offset of the meander, rotation center, angle of rotation of the external axis each moment, etc.