UNIVERSIDADE D
COIMBRA

Francisco José Ribeiro Ascenso

DESENVOLVIMENTO DE UMA ESTRATÉGIA PARA A DEFINIÇÃO DA ORIENTAÇÃO DE UMA FERRAMENTA DE DEPOSIÇÃO EM TAREFAS DE IMPRESSÃO 3D

SIMULAÇÃO EM MATLAB E REALIZAÇÃO EM ROBOTSTUDIO

VOLUME 1

Dissertação no âmbito do Mestrado integrado em Engenharia Mecânica, na área de Produção e Projeto, orientada pelo Professor Doutor Joaquim Norberto Cardoso Pires da Silva e apresentada ao Departamento de Engenharia Mecânica na Faculdade de Ciências e Tecnologias.

Outubro de 2020

1 2 9 0

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D COIMBRA

Development of a strategy to define the orientation of a deposition tool in 3D printing tasks - simulation in Matlab and realization in RobotStudio

Submitted in Partial Fulfilment of the Requirements for the Degree of Master in Mechanical Engineering in the speciality of Production and Project

Desenvolvimento de uma estratégia para a definição da orientação de uma ferramenta de deposição em tarefas de impressão 3D

Author

Francisco José Ribeiro Ascenso

Advisor

Professor Doutor Joaquim Norberto Cardoso Pires da Silva

Jury

President	**Professor Doutor Ricardo Nuno Madeira Soares Branco** **Professor Auxiliar da Universidade de Coimbra**
Vowels	**Professor Doutor Trayana Stoykova Tankova** **Professor Auxiliar da Universidade de Coimbra**
	Professor Doutor Carlos Xavier Pais Viegas **Professor Convidado da Universidade de Coimbra**
	Professor Doutor António Fernando Macedo Ribeiro **Professor Auxiliar da Universidade do Minho**
Advisor	**Professor Doutor Joaquim Norberto Cardoso Pires Silva** **Professor Doutor c/ agregação da Universidade de Coimbra**

Coimbra, October, 2020

“When you change the way you look at things, the things you look at change”

-**Max Planck**

ACKNOWLEDGEMENTS

Venho por este meio deixar um especial agradecimento:

Ao Professor Doutor Joaquim Norberto Cardoso Pires da Silva pela oportunidade, apoio e orientação ao longo da realização desta tese.

Ao Carlos Ye Zhu, pela partilha de conhecimento, disponibilidade e companheirismo.

Aos meus pais, por todo o esforço que fizeram ao longo do meu percurso académico. Pelo suporte e educação que me permitiram fazer as escolhas certas.

À minha namorada, por ter sido o meu principal pilar ao longo da minha formação como engenheiro mecânico.

À minha irmã, por todos os desabafos e bons momentos.

Aos meus avós, por terem sido a maior fonte de motivação ao longo destes cinco anos.

Ao meu colega Tiago Marques, por todo o apoio. Sem ele não seria possível realizar esta tese.

A todos os meus colegas que me deram suporte direta e indiretamente ao longo do curso.

Um grande Obrigado.

Abstract

This thesis integrates a project developed at the University of Coimbra by the Additive Manufacturing Constructions (AMC) group that intends to automate a robotic cell for Metallic Additive Manufacturing.

This work presents a possible solution to improve the concept of 3D metallic printing when applied to an industrial robot. Printing large parts and complex geometries remains a challenge for the additive manufacturing industry. The printing process of this type of part lacks automation, explained by the need for expert welding knowledge along with an algorithm to control the deposition geometry(Kazanas et al. 2012). Compared to a traditional method, additive manufacturing saves material by optimizing geometry. However, depending on the printed component there is often a need to add support material.

Thus, an algorithm has been created that makes the printing process more efficient, reducing total printing time and wasted material by reducing the need for support material, through a procedure that changes the printing orientation based on the geometry of the part to be manufactured.

A G-code analyzer was built in C# programming language to collect information about the geometry of the part and calculate the best printing orientation for each case.

To guarantee that the algorithm works properly, a robotic station which simulates the behavior of a real robot was created in RobotStudio. This allows the validation and verification of the algorithm, while singling out the improvements needed. Also, a program in Matlab was developed to verify and graphically represent most of the operations performed.

The whole procedure considers the use of WAAM printing technology, an ABB IRB 4600 robotic arm, an IRBT 2005 linear track, and the TPS 400i CMT Advanced Fronius printing tool.

Keywords 3D Printing, Industrial Robot, Orientation, C#, Simulation, RobotStudio, Matlab.

Resumo

Esta tese integra um projecto desenvolvido na Universidade de Coimbra pelo grupo AMC (Additive Manufacturing Constructions) que pretende automatizar uma célula robótica para Manufatura Aditiva de Metais.

Este trabalho apresenta uma solução possível para melhorar o conceito de impressão metálica 3D, quando aplicada a um robô industrial. A impressão de peças de grandes dimensões e geometrias complexas continua a ser um desafio para a indústria de produção aditiva. O processo de impressão deste tipo de peça carece em automação, justificado pela necessida de aplicar conhecimentos especializados em soldadura, juntamento com um algoritmo que controle a deposição(Kazanas et al. 2012). Em comparação com métodos tradicionais, a manufatura aditiva desperdiça menos material. No entanto, dependendo do componente impresso, há frequentemente a necessidade de adicionar material de suporte. Assim, foi criado um algoritmo que altera a orientação de impressão com base na geometria da peça, tornando o processo de impressão mais eficiente, reduzindo o tempo total de impressão e o desperdício de material, através da redução da necessidade de material de suporte.

Foi construído um programa que analisa G-code, na linguagem de programação C#, para recolher informação sobre a geometria da peça e calcular a melhor orientação de impressão para cada caso.

Para garantir que o algoritmo funciona correctamente, foi criada uma estação robótica no RobotStudio, que simula o comportamento de um robô real, permitindo verificar o funcionamento do algoritmo e identificar possíveis melhorias, bem como, um programa em Matlab que permitiu verificar e representar graficamente a maioria das operações realizadas.

Todo o procedimento considera a utilização da tecnologia de impressão WAAM, um braço robotico ABB IRB 4600, um eixo linear IRBT 2005, e a ferramenta de impressão TPS 400i CMT Advanced Fronius.

Palavras-chave: Impressão 3D, Robô Industrial, Orientação, C#, Simulação, RobotStudio, Matlab.

Contents

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS/ ABBREVIATIONS

List of Symbols

$\boldsymbol{x}$ – X coordinate

$\boldsymbol{y}$ – Y coordinate

$\boldsymbol{z}$ – Z coordinate

$\boldsymbol{Rx}$ – Rotation matrix X axis

$\boldsymbol{Ry}$ – Rotation matrix Y axis

$\boldsymbol{Rz}$ – Rotation matrix Z axis

$\boldsymbol{R}$ – Rotation matrix

$\boldsymbol{\Phi}$ – Roll angle

$\boldsymbol{\theta}$ – Pitch angle

$\boldsymbol{\psi}$ – Yaw angle

Acronyms/Abbreviations

MAM – Metallic Additive Manufacturing

WAAM – Wired Arc Additive Manufacturing

AM – Additive Manufacturing

TCP – Tool Center Point

IDC – International Data Corporation

ASTM – American Society for Testing and Materials

IEDI – Institute of Study for the Development of Industry

CAD – Computer Aided Design

CNC – Computerized Numerical Control

FDM – Fused Deposition Modeling

SLA – Stereolithography

DED- Direct Enery Deposition

1. INTRODUCTION

As is currently known, it is possible to divide the evolutionary process of the industry into 4 revolution periods.

We are now in a period of transition to the fourth industrial revolution, which can be characterized by the appearance of cyber-physical systems, which attribute new capabilities to both the routine of the population and the industry. It is a period when technology is part of the life of societies and even human life. Some of the concepts that have emerged at the industrial level are artificial intelligence, smart factories, and cloud computing. Amplifying some topics covered throughout this thesis, such as the growing branch of robotics and 3D printing(Kasinski et al. 2005).

Metallic Additive Manufacturing (MAM) is now widely accepted as a new paradigm for the design and production of high-performance components for aerospace, medical, and automotive applications (DebRoy et al. 2018).

3D printing appeared as a promising technology for the industry in various sectors such as(Mellor, Hao, and Zhang 2014):

- Prototyping, by the drastic increase of prototyping speed, introducing the concept of Rapid Prototyping;
- Design, with its custom versatility;
- Economic, being profitable with low volume productions

Nowadays, 3D printing is considered to be one of the 4.0 Industry pillars(Kasinski et al. 2005). However, there are several problems yet to be solved, the size of the parts is often not compatible with industrial requirements, 3D printed metal pieces have a high level of imperfections such as residual tensions, distortion and low adaptability for different geometries(DebRoy et al. 2018). By default, the printing process occurs vertically to the printing table, layer-by-layer. Through this procedure, it is common to achieve situations where extruded material cannot stand by itself, caused by the lack of a structure to deposit it, or situations where the weight and geometry of the piece do not allow it to remain stable. Therefore, the final piece has a step-shaped surface, causing high roughness and excessive material due to support material addition. In some cases, additional

machining to smooth the surface is required (Wu et al. 2018). As a result, MAM construction technologies are difficult to automate

For this purpose, a printing tool is connected to a 6-axis robot arm, allowing to increase the automation of the process by amplifying the workspace in a way that larger pieces can be printed, avoiding the necessity to separate the printing process into several pieces, being able to customize the printing process.

This thesis focus on adapting the 3D printing process of regular polymeric 3D printers to a robot system, contributing to improve MAM solutions.

The implementation of an automated system that enables the robot to adjust the printing tool orientation allows improving the printing process by reducing/removing the need for support material and improving the surface finish. (Kazanas et al. 2012). Both improvements lead to a reduction in printing and surface machining time. Through orienting the printing tool it is possible to direct the extruded material to the deposited material from the layers beneath. This way, the process always guarantees support for the new layers, and a reduction in support material is expected. The orientation algorithm is also thought to improve the surface finishing, keeping the printing tool parallel to the surface it is printing. Thus, the process becomes more efficient by minimizing the wasted material, reducing the printing process time, and improving the surface finish

To fulfill the objective of this thesis, an algorithm to orientate the printing tool, based on the aspects mentioned before was built. In partnership with my college Tiago Marques, the tasks to implement the algorithm were divided. While Tiago´s thesis focuses on implementing the algorithm through a programming language, this thesis focus on guiding the algorithm through: identifying needs, testing, detecting improvements to be made and simulate all the process.

1.1. Structure

This thesis was structured in the following order:

- **Chapter 1**- Introduces the theme of the thesis, giving a global view of the work done while emphasizing its potential and importance.
- **Chapter 2** - This chapter summarizes the information needed to understand the theme of this thesis. It presents the different types of 3D metallic printing, some characteristics of metallic printing, capabilities, and some examples of its application relevant to this thesis.
- **Chapter 3** - Brief description of the software used during this work.
- **Chapter 4** - This chapter presents the algorithm created to perform the orientation of the printing tool.
- **Chapter 5** - This chapter explains how the orientation calculated in the previous chapter was implemented within the RobotStudio simulation interface.
- **Chapter 6** - Description of the testing phase, which followed all the development of the orientation algorithm, allowing the implementation of several improvements.
- **Chapter 7** - Discussion of the results obtained based on two strategically chosen pieces.
- **Chapter 8** - Concluding this thesis, this chapter exposes the main conclusions and possible future works that can be done with this methodology as a base.

1.2. Motivation

With the growing competition in the Industrial Market, the search for a final product with a lower cost and an acceptable quality over the last years has driven the mass production sector to migrate its activities to developing countries, mainly to Asia(Mellor, Hao, and Zhang 2014).

The lower manufacturing and labor cost of this type of countries made Europe, and the majority of the developed countries, uncompetitive, creating an Industrial gap(Mellor, Hao, and Zhang 2014).

There is now, a necessity to explore new forms of business, which are not based in the old modules, developing a new market strategy focused on low volume productions with high-level quality and customization. Thus, as mass production began its migration, many technologies started to emerge, creating a revolution on the industrial branch.

One of these technologies is 3D Printing, officially created by Chuck Hull and his team, in 1984, 3D printing integrates a vast group, named Additive Manufacturing (AM), according to ASTM (American Society for Testing and Materials), it is defined as technologies that are based on a geometrical representation, creating physical objects by successive addition of material. (ASTM ISO/ASTM 52900-2015).

To understand the impact of 3D printing nowadays, according to IDC (International Data Corporation), in 2019, nearly 13.8 billion american dollars were spent in 3D printing, a value 22.1% higher than 2018. By 2022 it is expected a total spending of 22.7 billion american dollars.

Its application is no longer solely turned to the discretive manufacturing sector, it has now a vast amount of other consumers, the main ones being: Healthcare Provider, Education, Professional Services, Consumers, as shown in the figure below (IDC n.d.).

The Institute of Study for the Development of Industry (IEDI) highlights 3D

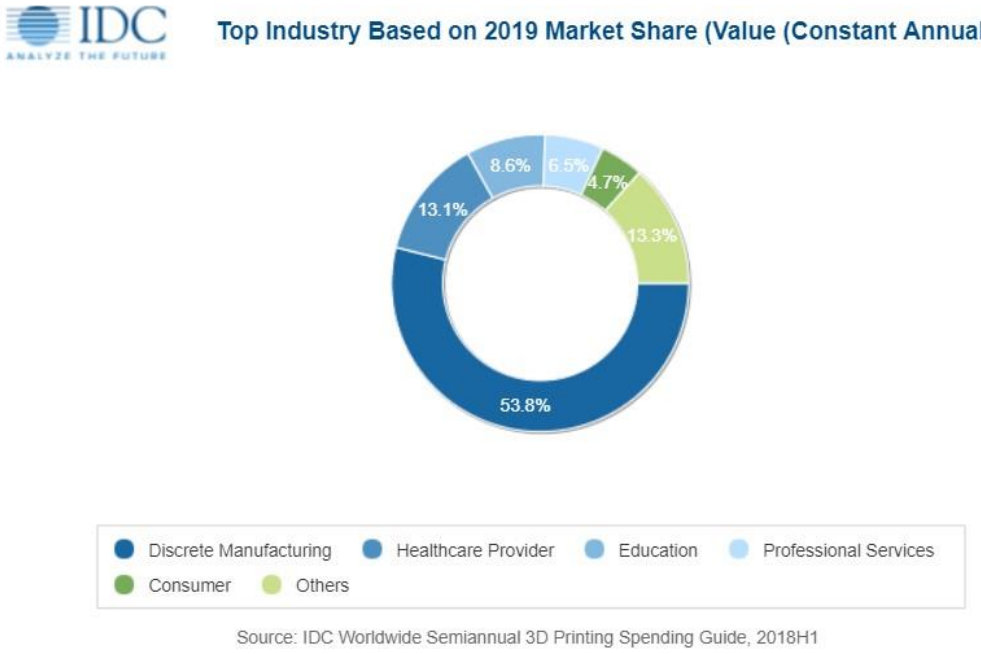

Figure 1.1 3D Printing Market Share (IEDI n.d.)

printing, considering that it is one of the most used and comprehensive technique, with a reduced waste of material and allowing rapid production on a small scale, which makes its development extremely important(IEDI n.d.).

1.3. Limitations

There are some limitations inherent to the method applied in this thesis.

The collection of the geometry data is based on the Verbose option of Slic3r version 1.3.0, which adds explanatory notes in g-code. This way, the algorithm does not work with other slicing programs, it is necessary to implement some adaptations.

Another limitation is that it is not possible to print with horizontal orientation since the orientation is calculated through points of different layers (with more detail in **Chapter 4**). Horizontal printing is useful to print cantilevered pieces, as so, the program cannot print cantilevered pieces.

1.4. Objetives

This thesis is part of a development project by the *University of Coimbra* regarding 3D metallic printing using a robotic arm. It aims to find a solution to the lack of automation in the field, as well as improving the process itself. As such it is a project of pioneering nature.

The main objective of this thesis is to optimize 3D metallic printing, minimizing the need for surface treatment of the part, after printing, as well as the need for support material, through a routine that punctually and automatically defines the printing orientation to be applied based on the geometry of the part to be manufactured.

All information related to the geometry of the workpieces is based on the traditional polymeric 3D printing procedure, so the entire procedure had to be adapted.

All the topics covered were discussed together with my colleague Tiago Marques, however, by division of tasks, Tiago will focus on the part of the implementation of the orientation algorithm in a programming language, whereas, this thesis is more focused on checking the operation and implementation of the orientation algorithm within an interface related to Robotics.

Thus, several tasks were defined:

- Obtain relevant data from the G-code;
- Build a visual module for verification;
- Calculate the rotations necessary to obtain an orientation parallel to the part´s surface;
- Apply orientation;
- Create a generic file;
- Implementation at RobotStudio;
- Improvements;
- Test;

2. STATE OF ART

In this chapter, a summary of the technology used to carry out a MAM (Metallic Addictive Manufacturing) project using a robotic arm will be presented, as well as the materials that can be used.

Several examples of the use of MAM technologies will be addressed, which help to realize the potential of this topic.

Finally, the material used and a global approach to the movement of an ABB robot will be exposed.

2.1. AM Technologies

ASTM has defined additive manufacturing (AM) as "a process of joining materials to make objects from 3D model data, usually layer upon layer"(Frazier 2014).

Initially, 3D printing emerged able to produce objects in polymeric materials, using stereolithography, a technology based on photochemical processes.

Over time, many new technologies appeared, according to the ASTM 52900-2015 standard, it is possible to identify seven process categories of 3D printing as shown in the figure below.

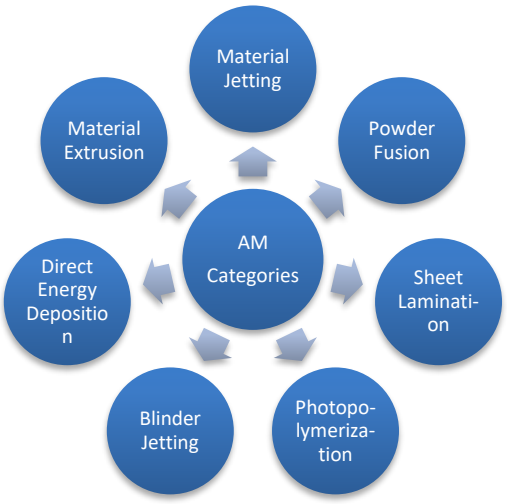

Figure 2.1 Classification of AM processes based on ISO/ASTM 529000:2015.

It is now possible to work, not only with polymeric materials but with metallic ones too, in the figure below is displayed a list of materials that can be used with 3D printing.

Table 1
Current commercial materials directly processed by AM, by AM process category.

	Amorphous	Semi-crystalline	Thermoset	Material extrusion	Vat polymerization	Material jetting	Powder bed fusion	Binder jetting	Sheet lamination	Directed energy deposition
ABS [Acryonitrile Butadiene Styrene]	X			X						
Polycarbonate	X			X						
PC/ABS Blend	X			X						
PLA [Polylactic Acid]	X			X						
Polyetherimide (PEI)]	X			X						
Acrylics			X		X	X				
Acrylates			X		X	X				
Epoxies			X		X	X				
Polyamide (Nylon) 11 and 12		X					X			
Neat		X					X			
Glass filled		X					X			
Carbon filled		X					X			
Metal (Al) filled		X					X			
Polymer bound	X	X		X						
Polystyrene	X						X			
Polypropylene		X					X			
Polyester ("Flex")							X			
Polyetheretherkeytone (PEEK)		X		X			X			
Thermoplastic polyurethane (Elastomer)				X			X			
Chocolate		X		X						
Paper									X	
Aluminum alloys							X	X	X	X
Co–Cr alloys							X	X		X
Gold							X			
Nickel alloys							X	X		X
Silver							X			
Stainless steel							X	X	X	X
Titanium, commercial purity							X	X	X	X
Ti–6Al–4V							X	X	X	X
Tool steel							X	X		X

Figure 2.2 Materials processed by AM

Source: Materials for Addictive Manufactoring 66(2017) 659-681

2.2. WAAM

In 1925, Baker proposed to use electric arc as the heat source with filler wires as feedstock materials to deposit metal ornaments and created an AM process called Wire Arc Addictive Manufacturing (WAAM)(Wu et al. 2018).

Most recently, many metallic AM technologies are being tested and improved, to maximize its efficiency, of which, WAAM stands out, as it is part of this thesis.

WAAM is a type of Direct Energy Deposition (DED) introduced in **Figure 2.1** and it is becoming one of the most promising AM technologies for various reasons, such as high material versatility, it is possible to reproduce a large number of metallic materials, such as steel, aluminum, nickel alloy and. titanium, a relative lower fabrication time and very

satisfactory mechanical properties(An Introduction to Wire Arc Additive Manufacturing [2020 Update] - AMFG n.d.).

A problem that has been pointed out for the use of this type of technology is the need to perform some surface treatment since the final appearance of the part is usually characterized by high roughness and some geometrical irregularities(Wu et al. 2018).

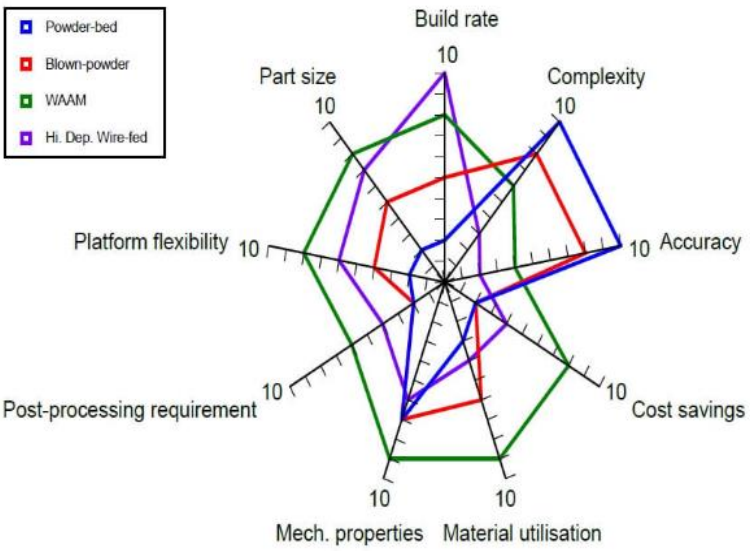

Figure 2.3 AM Properties
Source: https://www.3dmpwire.com/project-details/goals-and-objectives/

2.3. Orientation Impact

Through adapting orientation, it is possible to improve several aspects that are usually a problem for traditional 3D Printers.

In some cases, when the build orientation (represented in **Figure 2.5**) varies from positive values to negative ones, the overhang is generated. Overhang are geometric shapes in a 3D model that extend outside and beyond the previous layer. The overhang angle is crucial to determine if a piece can be printed with or without support material, it can increase with the height of the layer and the horizontal distance from the layer below. If the overhang is too high the printed material does not have enough support to maintain its position. To solve this problem, the support material is added. Support material has two purposes in the

printing process: it provides the necessary structural integration to prevent suspended parts from breaking and create a structure to deposit the extruded material(Jiang et al. 2018).

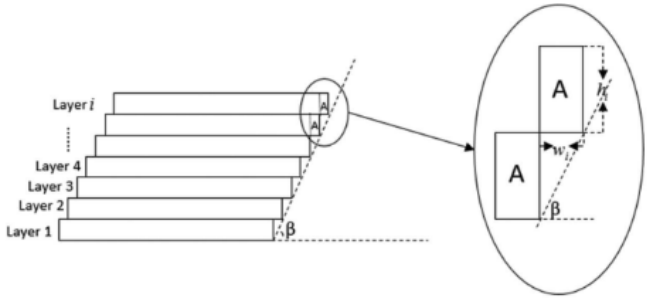

Figure 2.4 Overhang (Jiang et al. 2018)

When it comes to parts made of polymeric materials, the removal of support material is usually easy. For metal parts, machining is necessary and by its nature, the process takes significant time. Also a known problem of 3D printing is the Staircase effect, the construction layer by layer tends to form this shape, which causes high surface roughness, as shown in the figure below(Mohan Pandey, Venkata Reddy, and Dhande 2003).

By adapting the orientation of the printing tool, it is possible to increase the

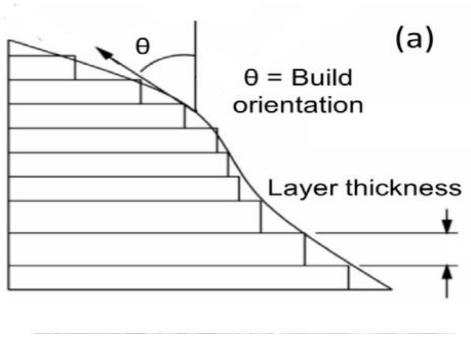

Figure 2.5 Staircase (Mohan Pandey, Venkata Reddy, and Dhande 2003)

contact of the overhanging material in 2 successive layers, minimizing the need for support material, causing the print tool to orient itself towards the extruded material in the previous layer, as shown in the figure below. This not only reduces machining time but also reduces production cost with less material needed and a possible smoothing effect of the stair shape effect, reducing surface roughness. All these improvements make the process faster and more sustainable. In 3D printing for metal parts, it has been proven that the decrease in printing time by improving the TCP speed is an incorrect approach, when changing the TCP speed, the properties of the weld bead become inconsistent(Kazanas et al. 2012). It is also possible to reduce the surface roughness by reducing the layer thickness, however, this

results in a drastic increase on the printing time. (Mohan Pandey, Venkata Reddy, and Dhande 2003)

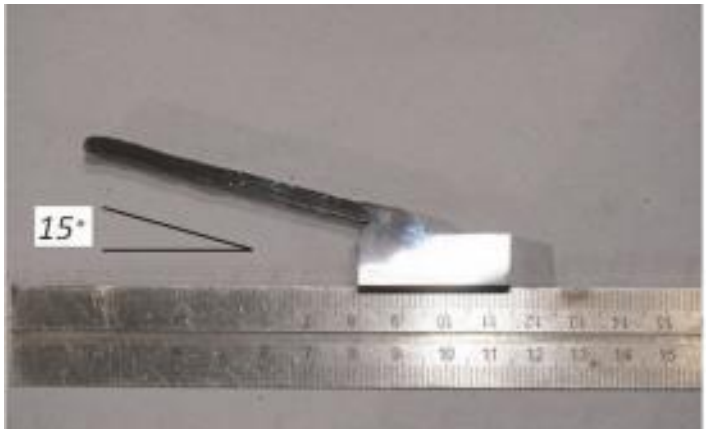

Figure 2.6 Inclined wall (Kazanas et al. 2012)

2.4. WAAM within robotics

There are several companies today that use WAAM technology for 3D printing. AM technologies have increasingly attracted the attention of the manufacturing sector of the industry due to the growing demand for a solution to print parts with good properties, large dimensions, complex geometry, and short processing time (Wu et al. 2018)

WAAM has the ability to work with several different metals, being in general one of the most economical processes in the 3D printing business, its mechanical properties are very comparable to those of the casting and forging process and when connected to a robot, the printed pieces size and flexibility increase significantly(An Introduction to Wire Arc Additive Manufacturing [2020 Update] - AMFG n.d.).

2.4.1. WAAM3D Container

As an example of the advantage of 3D printing, Cranfield University used WAAM technologies to manufacture the Thales Alenia Space pressure vessel. By switching from the conventional subtractive machining method to WAAM using Ti-6AL-4V as deposition material, WAAM3D printed the 1m high, 8.5kg container with a saving of 200kg of material and improved manufacturing time by 65% without giving up the required performances(Titanium pressure vessel for space exploration made using 3D printing n.d.).

Figure 2.7 3D Printer Container(Cranfield University) Source : https://www.pesmedia.com/titanium-pressure-vessel-space-exploration-additive-manufacturing/)

2.4.2. MX3D Bridge

Another example of the use of WAAM with a robotic arm using different orientations for parts production was the construction of a bridge. In 2015, the company MX3D launched a pioneering project in 3D metallic printing, proposing the printing of the first full-scale bridge, in-situ. The initiative was so successful that a considerable number of large companies in the industry wanted to be involved in the project.

Figure 2.8 Real Scale Bridge (MX3D Bridge | MX3D n.d.)

The challenge of building a large piece led to the company having to implement a robot movement system as the bridge was printed.

Since the initiative aimed at in-situ printing, MX3D had to find a solution that would allow printing the bridge, without the possibility of using a surface to support the weight of the part itself, as well as the use of support material.

The possibility to use in certain situations the printing on the horizontal arose, being that the printing layers would be positioned perpendicularly to the others, in this way, the deposition of the layers would be over the own part, annulling the problem related with gravity, while in a normal situation it would be necessary to use support material.

In 2016, the company concluded that horizontal printing is possible and applicable, having thus implemented this method for the realization of the bridge.

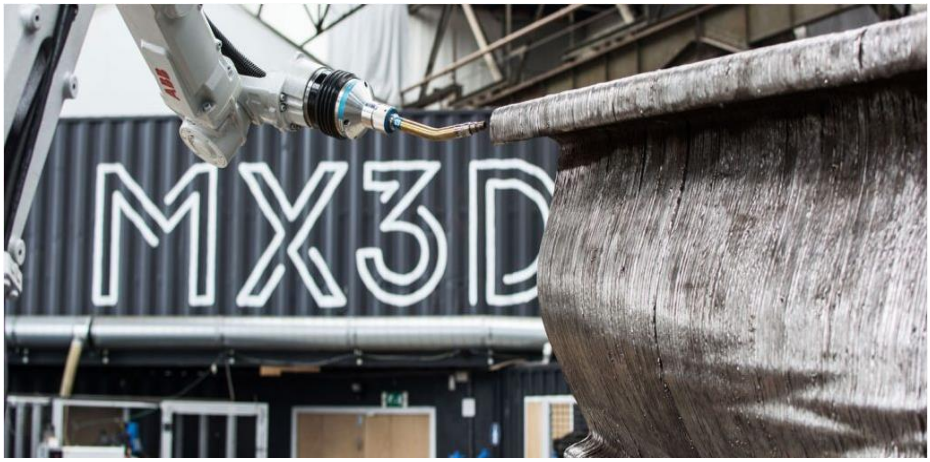

Figure 2.9 Horizontal Printing (MX3D Bridge | MX3D n.d.)

Finally, in 2018, the entire bridge was load-tested, obtaining affirmative results and consequently, its installation was approved.

With the printing of this bridge, it was possible to verify that it is possible not to use support material, in situations that normally would be applied, as well as, the obtaining of good results varying the orientation of the printing tool, when WAAM technology is used.

As previously mentioned and visible in the image above, this type of printing technology requires some kind of surface treatment, otherwise, the final aspect of the part will not be the most indicated(MX3D Bridge | MX3D n.d.).

2.4.3. MX3D Connector

Takenaka, one of Japan's largest architectural, engineering and construction companies, has designed a structural steel connector with the partnership of MX3D. This project shows the progress in the production of highly customized steel connectors, designed using 3D metallic printing on robots equipped with Wire Arc Additive Manufacturing (WAAM) technologies. The net weight of the steel connector is 40Kg, reaching 45Kg after filling with mortar. The MX3D printed this connector using Duplex stainless steel, an alloy known for its good mechanical properties(TAKENAKA CONNECTOR | MX3D n.d.).

Figure 2.11 Takenaka Connector (TAKENAKA CONNECTOR | MX3D n.d.).

Topology optimization was used to produce this connector allowing full digital control over the design, production, timeline, and cost with great logistical benefits(SOFTWARE | MX3D n.d.).

As it is shown in **Figure 2.11** topology optimization tends to obtain complex geometries. These pieces are one example that the method presented in this thesis is thought for.

2.5. Station

The station is composed of three main components:

- ABB IRB 4600 Robot ARM
- IRBT 2005 linear track
- TPS 400i CMT Advanced Fronius printing tool.

The figures below show the comparison between the real robotic station and the virtual version used to perform the tests.

Figure 2.12 Real Robotic Station

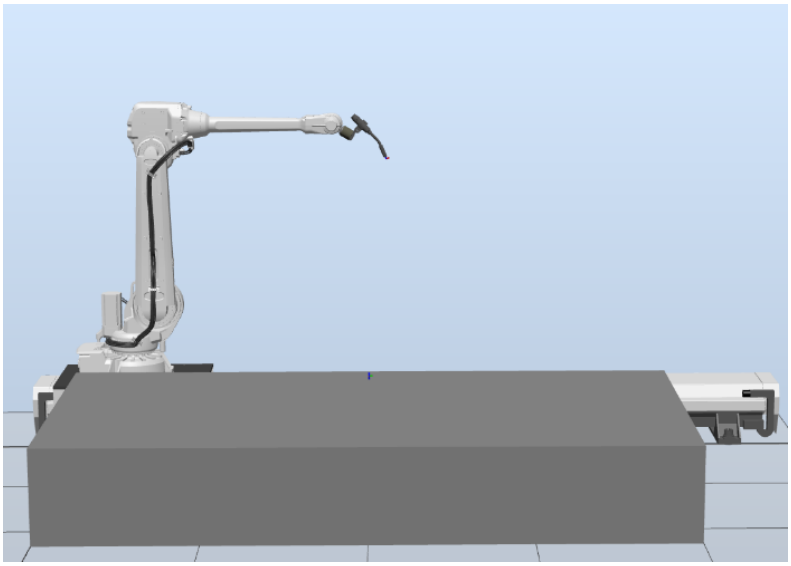

Figure 2.13 Virutal Robotic Station

2.6. Robot movement

In the operation of industrial robots, it is necessary to describe the relative position of several components in order to not cause damage and to guarantee the good functioning of the system. Being the main components, the robot tool, surrounding objects, the robot joints and links.

For this purpose, each component is associated with a reference system that correlates the relative position and orientation between that component and the others.

Position vectors are used to define the relative position of an object, while rotation matrices are used to define the orientation.

The structure of an industrial robot is most of the time made up of rigid links interconnected by six joints, as shown in the figure below.

The position of the joints together with the dimension of the links defines the tool position (Pires 2020).

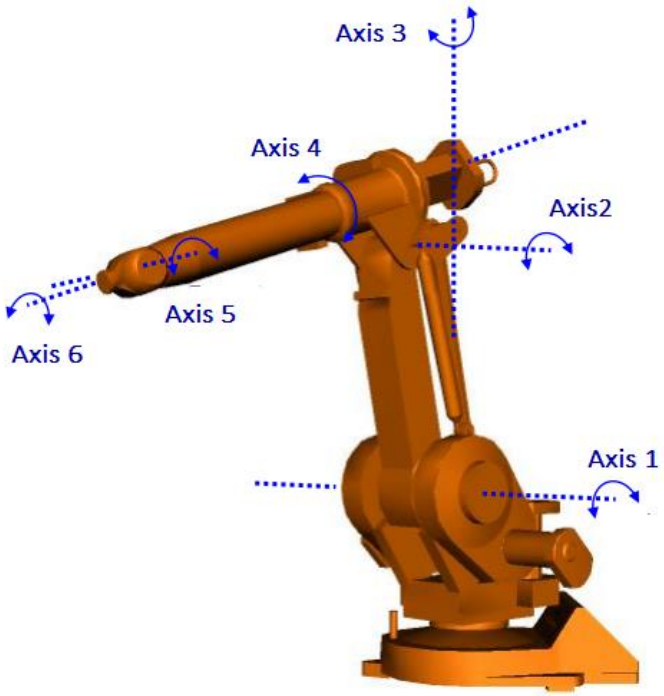

Figure 2.14 Industrial Robot Axis, adapted from (Pires 2020).

RobotStudio's interface allows to define 3 movement types for TCP (Tool Center Point): Linear, the TCP moves linearly between 2 points, Circular the TCP moves circularly between 2 points, and Joint the TCP moves quickly between 2 points, not respecting movement types.

To define a position within the RobotStudio environment (simulation and control program and make the robot move to that position, according to the Rapid (programming language related to robotstudio) manual, it is necessary to define a robtarget, the variable used to define the Tool Center Point (TCP) position, and assign one of the three movement types described above. This variable is composed of 4 components, represented in the figure below(ABB 2018):

```
robtarget Target_1:=[[x,y,z],[q1,q2,q3,q4],[cf1,cf4,cf6,cfx],[e1,e2,9E+09,9E+09,9E+09,9E+09]]
```

Figure 2.15 RobTarget

2.6.1. Translation:

Cartesian position (x,y,z) of the TCP expressed in millimetres. It is specified in relation to the current WorkObject (cartesian referential).

2.6.2. Rotation:

Defines the orientation of the tool, expressed in the form of a quaternion (q1,q2,q3,q4), as shown in **Figure 2.12**, or Euler angles.

This thesis uses Euler Angles to define the rotations of the robot. This way, it is important to understand how Euler angles work.

In terms of the order, there are two types of classification:

1.**Tait-Bryan angles**

In this approach, 3 rotation axes (x,y,z) are used to define the global rotation of the system.

The order of rotation of the axes may be varied according to 6 combinations and the following configurations are possible: (x-y-z ; x-z-y ; y-x-z ; y-z-x ; z-x-y ; z-y-z).

2.**Proper Euler angles**

In this approach, only 2 rotation axes are used to define the overall rotation of the system, one of which is repeated.

There are also 6 possible combinations: (x-y-x ; x-z-x ; y-x-y ; z-x-z ; z-y-z).

As for the convention, there are two types of classification:

1. **Extrinsic**

The original coordinate axis stays stationary and the coordinate axis of the system moves relative to the first one.

2. **Intrinsic**

The coordinate axis of the system is rotated according to itself, which means that each rotation around a certain axis is based on the rotated axis and not on the fixed one(Diebel 2006).

2.6.3. Robot configuration:

Axis configuration of the robot, indicates the quarter revolution of the robot on axis 1, axis 4 and axis 6 (cf1,cf4,cf6) and cfx is dependent on robot type

2.6.4. External axes

Position of the external axes. A robot can have multiple external axes, the position of each is defined on the e1, e2, etc. If, for example, the robot only has one external axis, only the e1 parameter will have a valid number, the “9E+09” represents the axes which are not connected.

3. SOFTWARE

Throughout the development of this thesis, it was necessary to resort to several engineering/programming tools to respond to the various tasks defined, which as a whole, allow the achievement of the previously mentioned objectives.

It is important to know the capacity of each program to understand its use throughout the description of the methodology.

In this way, the programs used are briefly presented, as well as their capacities.

3.1. Slic3r

According to its own creators and developers, slic3r is classified as an open-source 3d printing toolbox(Slic3r n.d.).

Its main function is to horizontally slice each test piece into homogeneous slices and convert this piece, generated in a CAD program, into G-code language.

Today there are several programs of this type that could have been chosen for use in this thesis. The choice of Slic3r is due to the following aspects:

- Visual preview of toolpaths
- Advanced configuration management
- Custom G-code with conditional logic
- Modifiers (distinct settings on custom regions)
- Free to use

3.2. G-code

Initially created to homogenize the CNC programming language, the G-code may be defined as a programming language whose purpose is to control machines.

At its core, the G-code is a text file, with a set of lines that contain information related to the movement of a machine.

Within the text file, there are two types of information, the execution lines, which are read by the controller and contain information for carrying out some type of movement by the equipment to be used, and documentation lines, which start with a special character and only serve to describe the process and are not executable(Champlain 2015).

In the case of the Slic3r, the G-code is designed to be used with Fused Deposition Modeling (FDM) and Stereolithography (SLA), two processes for printing polymers, normally applied in traditional printers, as such, will have to be properly interpreted to function with the System related to this thesis.

The figure below shows an example of a G-code.

```
G1 Z0.350 F7800.000 ; move to next layer (0)
G1 E-2.00000 F2400.00000 ; retract extruder 0
G92 E0 ; reset extrusion distance
G1 X90.240 Y46.026 F7800.000 ; move to first skirt point
G1 E2.00000 F2400.00000 ; unretract extruder 0
G1 F1800
G1 X91.550 Y44.144 E2.07096 ; skirt
G1 X93.424 Y42.821 E2.14192 ; skirt
G1 X96.251 Y42.188 E2.23157 ; skirt

G1 E8.63541 F2400.00000 ; retract extruder 0
G92 E0 ; reset extrusion distance
G1 X123.252 Y98.073 F7800.000 ; move to first perimeter point
G1 E2.00000 F2400.00000 ; unretract extruder 0
G1 F1800
G1 X123.332 Y100.000 E2.05967 ; perimeter
G1 X123.252 Y101.927 E2.11933 ; perimeter
G1 X123.872 Y97.996 F7800.000 ; move to first perimeter point
G1 F1800
G1 X123.957 Y100.000 E6.59436 ; perimeter
G1 X123.875 Y101.978 E6.65563 ; perimeter
G1 X124.492 Y97.919 F7800.000 ; move to first perimeter point
G1 F1800
G1 X124.583 Y100.000 E11.25061 ; perimeter
G1 X124.499 Y102.030 E11.31347 ; perimeter
G1 X124.451 Y97.950 F7800.000 ; move inwards before travel
G1 E13.96155 F2400.00000 ; retract extruder 0
G92 E0 ; reset extrusion distance
G1 X115.135 Y116.949 F7800.000 ; move to first infill point
G1 E2.00000 F2400.00000 ; unretract extruder 0
G1 F1800
G1 X116.949 Y115.135 E2.08008 ; infill
G1 X118.161 Y113.703 E2.13860 ; infill
G1 Z0.350 F7800.000 ; move to next layer (1)
G1 E83.80703 F2400.00000 ; retract extruder 0
G92 E0 ; reset extrusion distance
G1 X103.749 Y73.613 F7800.000 ; move to first support material point
G1 E2.00000 F2400.00000 ; unretract extruder 0
G1 F1800
G1 X103.749 Y65.337 E2.25604 ; support material
G1 X102.500 Y65.337 E2.29471 ; support material
G1 X102.500 Y73.613 E2.55076 ; support material
G1 X101.250 Y73.613 E2.58943 ; support material
G1 X101.250 Y65.337 E2.84547 ; support material
```

Figure 3.1 G-code example

To help identify and understand the information, the Verbose Slic3r 1.3.0 G-code functionality was deployed, it adds detailed information of the function of each line, after the ";" character, thus being possible to differentiate the types of points, as well as to understand what the order of printing of them is, as it is represented in **Figure 3.1**.

It is possible to verify that there are 5 types of points:

- Skirt Points
- Brim Points
- Perimeter Points
- Infill Points
- Support Material Points

The skirt and brim points are always the first to be printed, their function is to guarantee that the temperature of the printing tool is with the thermal and debit conditions indicated before starting the printing of the part itself, guaranteeing that there is a good adhesion of the part to the printing table, avoiding slipping and other damages. In the picture below it is possible to identify the skirt and the brim by the green color and the yellow line, outlining the component.

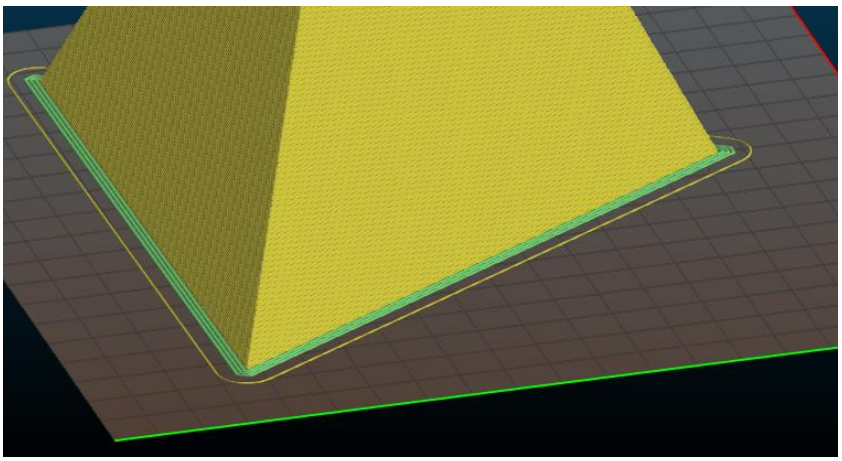

Figure 3.2 Skirt and Brim Material

The Perimeter, Infill ,and Support Material points are printed in this order and are the points that together form the piece to print.

As the name implies, the perimeter points are points that constitute the perimeter of the piece, or its periphery, the infill points, constitute the whole filling of the piece while the support material has an auxiliary role in the printing process.

In the figure below it is possible to identify the three types of points mentioned above.

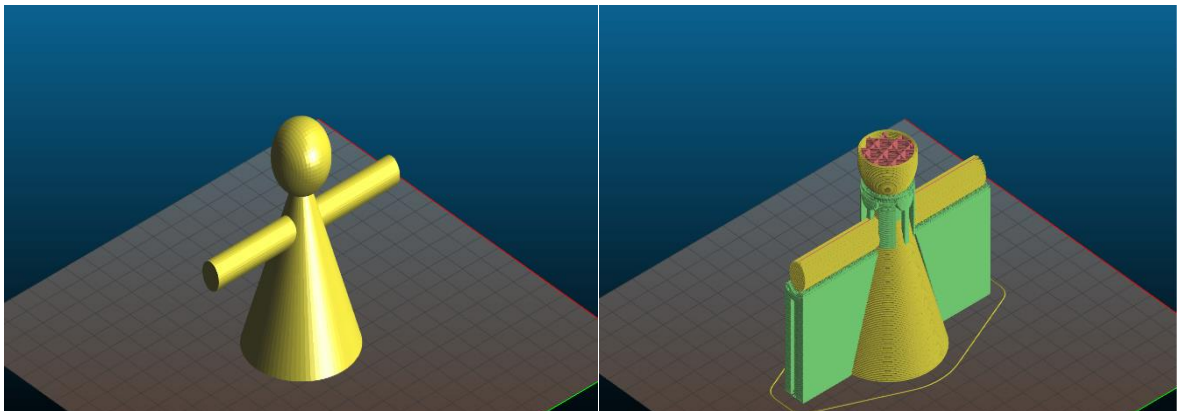

Figure 3.3 Infill Support and Perimeter Material

In **Figure 3.1** several commands of type G1, G92, etc are visible. These are related to actions such as print start, print tool retract, and others. At this stage, they are not important information, because the printing system is not a common 3D printer, but an industrial Robot, as such, these actions have to be thought and adapted in another way and on another platform, such as robotstudio.

3.3. C#

C# is an object-oriented programming language, developed by Microsoft, integrating the .NET platform.

Since this subject is full of innovative content, it is essential to use software with the ability to create robust applications with high versatility.

Thus, through C#, it was possible to make a totally original algorithm, which compiles the information obtained through the use of the software mentioned and returns the necessary information for the orientation of a robot.

3.4. Matlab

Matlab is one of the most powerful numerical calculation software as of today, it is commonly used for numerical analysis, graph construction, matrix calculations, or other numerical operations and graphic representation(MATLAB n.d.)

In the context of this thesis, Matlab contributes essentially to a fast verification in the form of graphical representations, allowing the visualization of operations performed in other interfaces.

3.5. RobotStudio

RobotStudio is a software developed by ABB, with the main objective of simulating and controlling an ABB robot through a digital interface(RobotStudio n.d.).

In RobotStudio it is possible to build a work station that accurately represents the behavior of the robot in reality, using precise measurements and its components, this concept is given the name of digital twin.

By the nature of this thesis, this platform is of extreme importance because it allows:

- Reduce the risk of damage
- Avoid unwanted stops
- Improve productivity
- Detect errors
- Detect Improvements

3.5.1. Rapid

The rapid is the RobotStudio intrinsic programming language, used to make a custom control of the robot.(ABB 2018)

It allows:

- User defined Functions
- Detect errors automatically
- Introduce Procedures
- Trap Routines
- Making arithmetic calculations
- Define variables

3.6. Inventor

Developed by Autodesk, the inventor is one of the greatest mechanical design software of today(Inventor | Mechanical Design & 3D CAD Software | Autodesk n.d.).

It allowed the design of all test pieces, as well as their conversion to a format that can be read by Slic3r

4. METHODOLOGY

Throughout this chapter, the entire procedure inherent to the customization of a Robot's tool orientation will be described.

4.1. G-code Data

For a robot to execute a certain position with a certain orientation, 4 components must be taken into account: translation, orientation, configuration, and external axis position. This task is limited to obtaining the values needed to fill the translation component and collecting information to define the rest.

Obviously, for a part to be printed by a robot, that part must be readable in the robot language. Thus, it is necessary to create a G-code interpretation method and return the information in a readable format to the robot controller.

The first step is to draw a part in the inventor interface or other drawing software and convert the part to a CAD format (.stl), extension readable in slicing programs.

Then it is imported into the Slic3r interface and the part is sliced.

Slic3r offers several customization options, such as the height of layers, the number of passages in the perimeter, the format of the infill of the part, the existence of material support, and the customization of G-code writing, among others.

At this stage, only the option of customization of the G-code writing, Verbose, is relevant, since it helps in the interpretation of the G-code. The remaining options do not significantly affect the results obtained in the data processing.

4.2. G-code interpretation

The slicing program divides the part into several layers, which in turn are divided into several points.

After slicing the previously drawn part, a text document, G-code, is generated. This document contains a list of ordered points that constitute the part to be printed, interspersed with movement instructions. It is important to mention that this information is relative to a part to be printed on a traditional polymeric material printer, as such, the printing parameters contained within the G-code have to be later adapted.

In this section, the process for obtaining the geometric coordinates that characterize each point of a part, after being processed in the Slic3r program, will be described.

An algorithm has been developed, with the help of Tiago, to organize and filter the necessary information.

In G-code, each point has inherent to itself 4 variables, 3 cartesian (X,Y,Z) being that X and Y are directly in the line that describes the point, whereas, Z is common to all points of the same layer and is represented at the beginning of each layer, there is also a variable relative to the extrusion of the material (E).

The variable E, does not need to be stored because it only counts the volume of extruded material and since this software is optimized for SLA and FDM, its values have to be adapted.

A C# methodology was created that interprets each G-code line and stores all points within three dictionaries (perimeter, infill, support material), where the key is the layer number and the values are the list of the points of the respective layer.

The identification of the points is made through a procedure that identifies the character "Z", that appears when there is a change of layer, then looks for the characters "X" and "Y" and stores this information in the dictionary relative to the perimeter, infill or support material, making its distinction through the interpretation of comments generated by the Verbose option.

Thus, all the information is available to complete the translation component of a robtarget, it is now necessary to build the methodology to fill the remaining components of the movement.

A Matlab visual test was performed to verify if the implemented algorithm is working properly and identifies all the perimeter points, and the results were positive, as shown in the figure below.

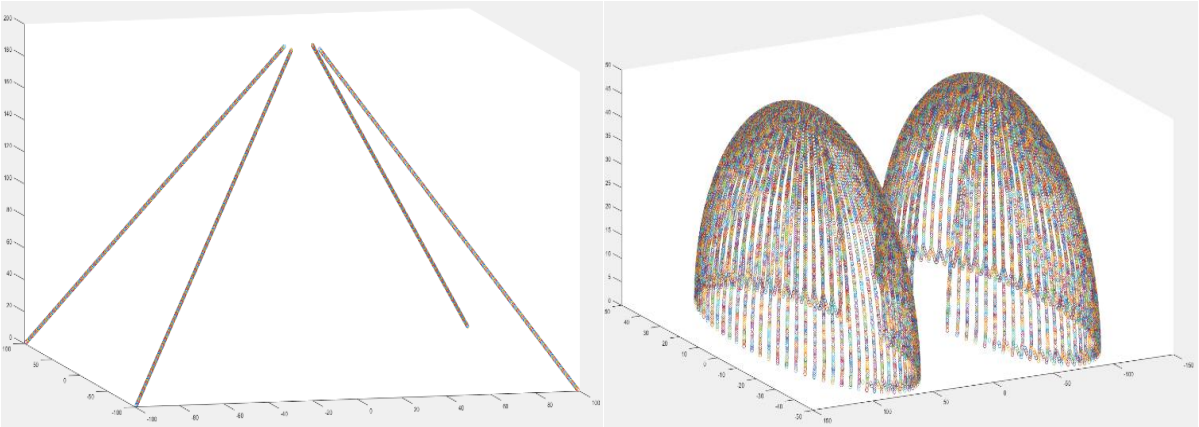

Figure 4.1 Matlab Preview

4.3. Orientation Algorithm

This chapter treats the main objective of this thesis, which is to obtain the second component of the robtarget, the orientation.

The perimeter points are the first to be printed so they are the most important for the integrity of the part.

In this way, the orientation will be determined according to the geometry of each part, for this, the necessary information to assign orientation is only the positional coordinates of the perimeter points. The infill points will be introduced in the printing process, however, with vertical orientation. The support material points, skirt, and brim do not need orientation because they are not part of the part itself, however, it is important to include them in the printing process.

The G-code interpretation methodology, introduced in **Chapter 4.2**, has gathered the necessary information to calculate the orientation of each point.

To determine the orientation of each point, a C# algorithm was constructed with Tiago. The algorithm input is the cartesian coordinates of each point and the output is the orientation of the corresponding point. Thus, the algorithm was based on the following method:

1. Separate the points by type

2. For Perimeter points, find the nearest point in the layer below. This task involves finding the 2 nearest points in the layer below and drawing a virtual line between them. Then, go through the virtual line and find the nearest possible location to the point to be analyzed and create a virtual point.

3. Pair the point under analysis with the virtual point.

4. Calculate the orientation of the point under analysis based on the coordinates of the virtual point. The orientation corresponds to the vector that points from the point under analysis to the virtual point.

It is necessary to know in advance that the tool orientation corresponds to the orientation of the Z-axis of the tool, represented in blue in the figure below, which in turn is calculated in relation to the WorkObject, represented in the corner of the table, as shown in the figure below.

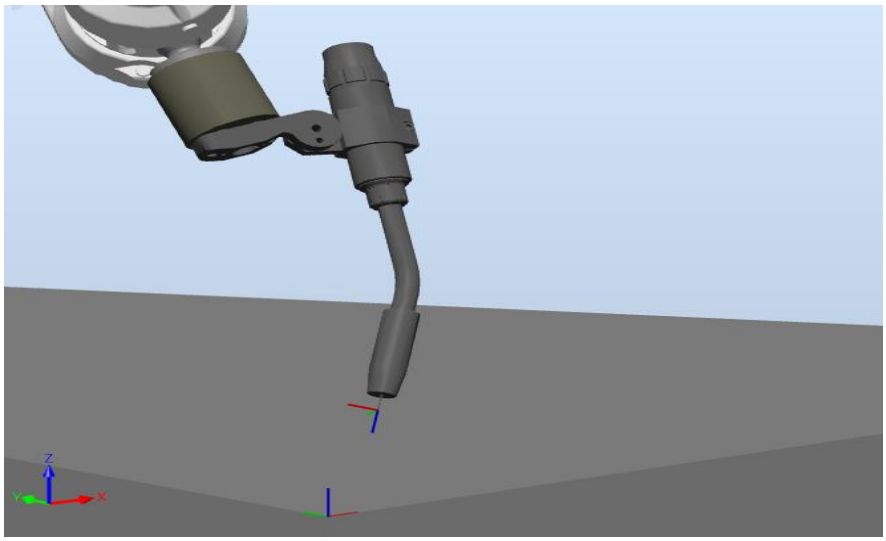

Figure 4.2 RobotStudio Orientation

For this purpose, Euler angles were used, whose function is: to define the rotation of any rigid object in three-dimensional space, through a fixed reference and a second that is rotated together with the moving object.

Euler angles can be classified according to the order of rotation and the convention used.

It is important to clarify which approach was followed to determine Euler angles, since, among the different options, the final result is not the same.

While all options are equally viable to solve the problem, this thesis uses Tait-Bryan order and Intrinsic convention to determine the Euler Angles.

Before implementing the method of calculating the rotation angles in the orientation algorithm, a verification test was performed, described in **Cap5.2**, to ensure that the entire process is correctly defined.

4.4. Calculating the orientation in Matlab

This chapter describes the process that originates the angles of rotation. Thus, a Matlab program has been realized that besides calculating the rotation angle, it also makes a graphical representation for verification purposes, to implement this method in the algorithm performed in C#.

As mentioned before, to define a rotation in three-dimensional space, it is necessary to use 3 rotation angles. As a rule, depending on the axis on which one is rotating, the angle has a certain name. Thus, for a rotation on the x-axis, the name attributed to the angle will be roll, a rotation on the y-axis will have the name pitch, and finally, on the z-axis, the name will be yaw.

It is necessary to find the correct roll, pitch, and yaw values for a rotation to be successful.

The way each angle has been calculated is quite trivial since it is only necessary to use trigonometric relations between the coordinates of the points that form the orientation vector(Diebel 2006).

The process begins by subtracting the X, Y, Z coordinates of the point closest to the layer below from the coordinates of the current point, points that were paired in **CAP 4.3.** Thus, the current point will be represented by the origin of the referential, while the point of the layer below is represented by a red circle, as shown in the figure below.

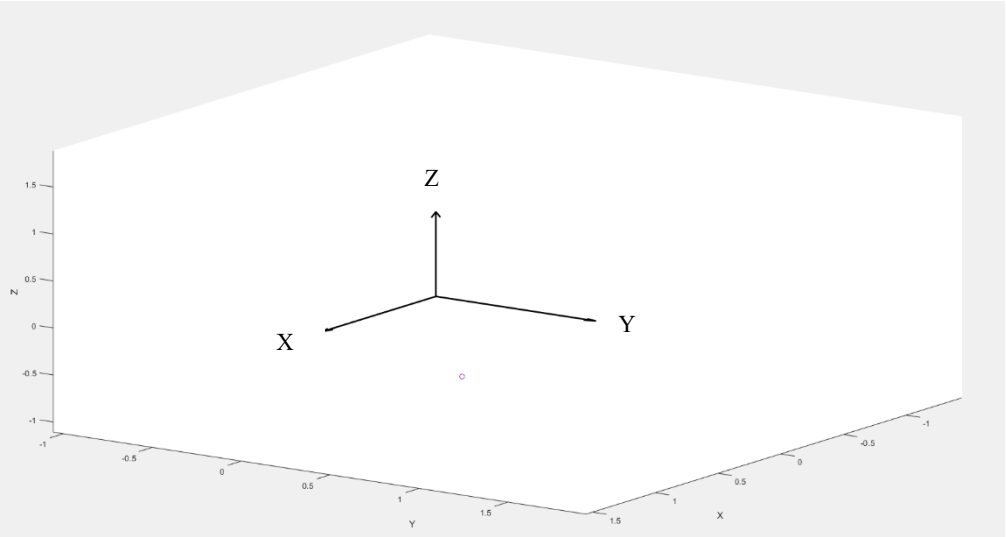

Figure 4.3 Original Referential with a point from the layer bellow

Note that the orientation of the robot must always have a negative direction relative to the original Z axis, otherwise the tool will be working under the table, which is not possible.

After subtracting the coordinates, the first angle (roll) is calculated as shown below.

$$roll = tan^{-1}(\frac{y}{z}) \tag{1}$$

After calculating the roll angle, to make a rotation around the X-axis it is necessary to multiply the vectors that constitute the original reference, by the X rotation matrix, Rx, whose formula is presented below.

$$Rx = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\Phi & -sin\,\Phi \\ 0 & sin\Phi & cos\Phi \end{bmatrix} \tag{2}$$

For verification purposes, the new reference is represented, now rotated Φ degrees on the x-axis. To make this representation it is enough to divide the matrix vectorially and represent the respective vectors, obtaining the following result:

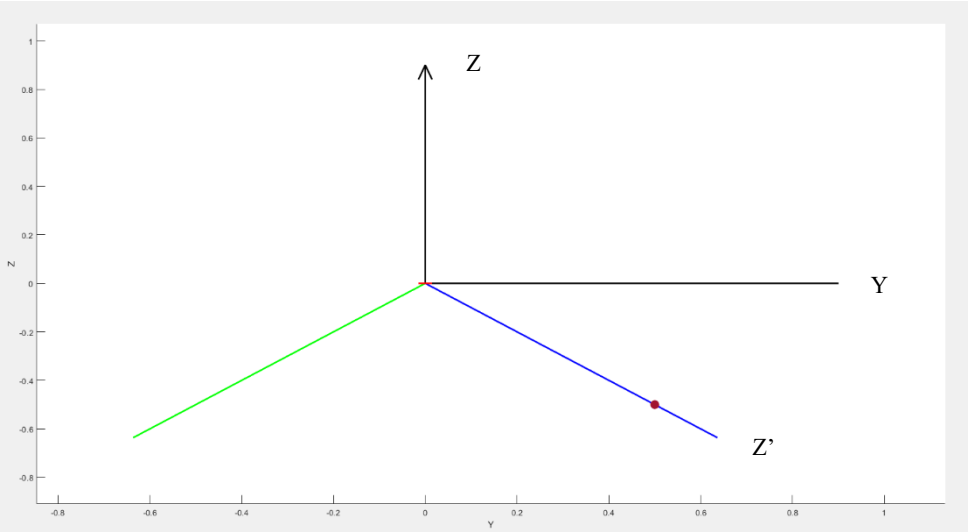

Figure 4.4 Y-Z View of the referential rotated on X axis

To comply with RobotStudio, the colors of the rotated axes correspond to the colors of the axes on Matlab, red x-axis, green y-axis, and blue z-axis. The fixed axis is represented in black. The x-axis is not visible since it is perpendicular to the view represented.

As can be seen in the figure above, after applying the rotation on the x-axis, one can conclude that the roll angle is well defined, the rotated z-axis is perfectly aligned with the orienting point.

The next step is to make the z-axis coincide with the orienting point to ensure that the tool has the desired final orientation.

To do this, it is necessary to rotate the reference axis again, now along the Y-axis. Thus, the pitch angle is calculated and defined trigonometrically by the equation below.

$$pitch = tan^{-1}(\frac{x}{\sqrt{y^2 + z^2}}) \quad (3)$$

The rotation matrix, according to Y, is now calculated with an angle of magnitude θ, using the formula presented below.

$$Ry = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\,\theta & 0 & cos\theta \end{bmatrix} \tag{4}$$

It also checks the rotation along the Y-axis, now waiting for the Z-axis to move to the orienting point. However, it is necessary to multiply the Rx matrix by the Ry matrix to obtain a combined rotation and respect the intrinsic convention mentioned above.

The following matrix is then obtained:

$$R = Rx * Ry = \begin{bmatrix} cos\theta & 0 & sin\theta \\ sin\theta sin\Phi & cos\Phi & -sin\,\Phi cos\theta \\ -sin\,\theta cos\Phi & sin\Phi & cos\Phi cos\theta \end{bmatrix} \tag{5}$$

Now proceeding to the representation of the new referential, dividing the matrix into vectors, as it was previously done, the result is presented in the figure below.

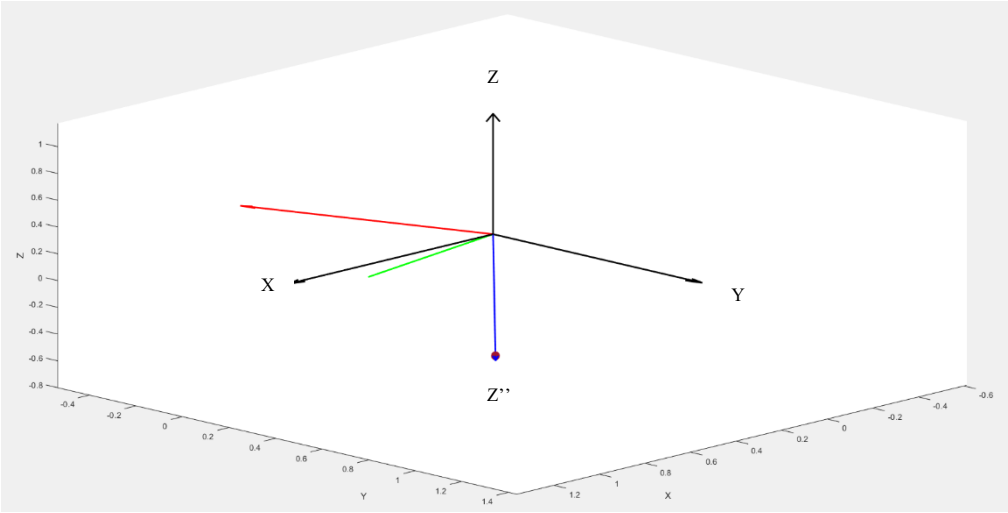

Figure 4.5 Isometric view of a full rotation

As expected, the combined rotation of Rx with Ry verifies that it is possible to orient the Z-axis according to the orienting point accurately.

Finally, the value of yaw may be ignored as the rotation around Z does not affect the spatial position of the Z-axis, only changing the positions of the X and Y axes, as shown in the matrix below.

$$Rz = \begin{bmatrix} cos\psi & -sin\,\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

This way, it is verified that the method is possible and correctly defined and can be implemented in the orientation algorithm.

5. COMMUNICATION WITH ROBOTSTUDIO

First, it was necessary to create a copy of the real system within the RobotStudio interface, to create a copy of the real system, a digital twin. The digital twin is a concept that can be seen as a bridge between the real world and the digital world, the accuracy between the model in RobotStudio and the real Station allows to perform tests, improvements and predictions about the behavior of the real system without causing damage to it.

With the digital twin consolidated, it is now necessary to send information regarding the operation of the same. There are several ways to meet this need, this thesis explores the exchange of information via text files (.txt), since it makes it easier to identify errors by analyzing them, and is a form of communication of simple understanding.

A program was created in Rapid with the main objective of reading the information contained in the text document.

The information in the text document comes in the following format:

T_;

P_ X_ Y_ Z_ OX_ OY_ OZ;

Each "**_**" delimits the information that is retained in the RobotStudio station. At the beginning of the document, the "**T**" type of orientation and movement to be used is indicated (this subject is dealt with in **Cap 6),** "P" is the point number, "**X**", "**Y**", "**Z**" are the coordinates of each point and finally "**OX**", "**OY**", "**OZ**" are the rotations around each axis.

The program identifies the characters in bold above and saves the values that succeed them. It then assigns these values to movement instructions, analyzing the text document line by line.

5.1. Commands

The first step for the part to be printed in the right place is to position WorkObject.

The WorkObject is the reference on which the robot tool moves, so the place where the WorkObject is positioned represents the origin of the movement and defines the direction of each of the X, Y, Z axes.

In this particular case, the workobject was centered in the surface of the table.Thus, if the part is to be centered with the table, it is necessary to place the reference in the center of the part on the interface of the Slic3r, since the reference of the Slic3r will coincide with the WorkObject.

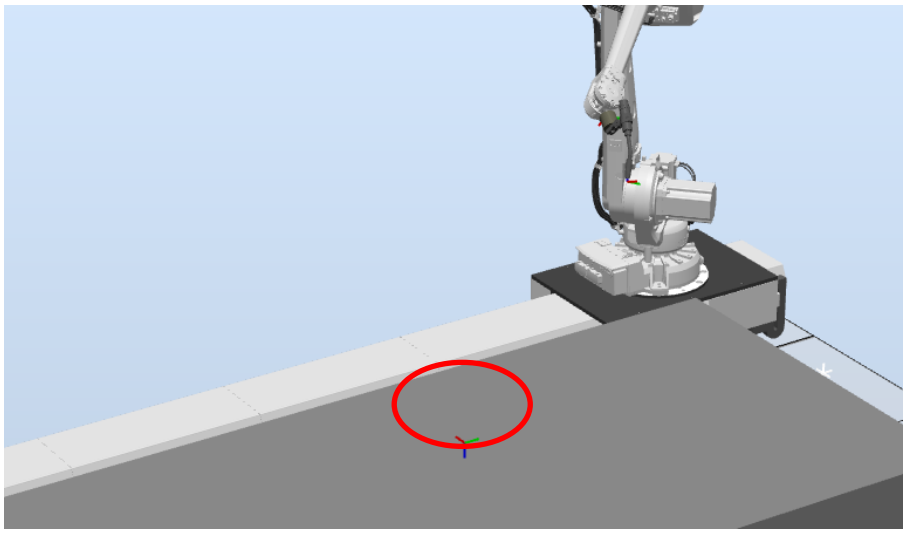

Figure 5.1 WorkObject

Once the WorkObject is defined, it is now possible to start the process of moving the robot.

There are 3 possibilities for the movement type of an ABB robot, namely linear movement, circular movement, and joint movement. As the G-code is built based on linear movements, this will be the implemented movement type, so it will be necessary to use the Rapid MoveL instruction.

In terms of spatial displacement, the function **Offs**, is used to add an offset in the object coordinate system to a robot position, according to the Rapid manual.

This function is based on robtargets, however, has the advantage that it is not necessary to create several points to move the arm. Only one point is assigned, and the movement is based only on instructions relating to it.

The offset will be made with reference to the WorkObject (point corresponds to the coordinate (0,0,0)).

In terms of angular displacement, the function **RelTool**, which is used to add a displacement and/or a rotation, expressed in the active tool coordinate system, to a robot position, according to the Rapid Manual,. will be used.

The work principals of this function are very similar to **Offs**. The difference is that **Offs** move in relation to the WorkObject and **RelTool** move in relation to the tool frame, having the possibility of introducing rotation angles.

At this point in the thesis, all the values related to orientation exist, it remains now to implement and work all that information together with the commands.

6. TESTING PHASE

During the testing phase with the RobtStudio interface, several problems were identified. This way, it was possible to improve the algorithm that defines the orientations, adding several functionalities and corrections.

6.1. Singularities Control

It is common in the field of robotics to implement functions that aim to protect the robot from reaching singularity conditions.

In the RobotStudio interface, it is also common to use these functions, which most of them, according to the rapid manual, defines how the robot is to move in the proximity of singular points.

There are two possibilities:

1. Apply the functions to all moves, which will cause the robot to lose some degrees of freedom in the movement or deviate from the pre-defined path to avoid singularity configurations.
2. Keep this function deactivated and only apply it, punctually.

The first hypothesis is not applicable, because it does not allow the expected trajectory and does not consistently define the orientation, it is necessary to use the second hypothesis, which is implemented through:

- Speed Control
- Maximum Angle
- Free Configuration
- Linear track movement
- Proximity to singularity points

6.2. Speed

Speed is a parameter that has to be evaluated before starting the printing process.

Besides interfering with the welding properties, during the translation, several axes of the robot need to move to ensure that the orientation corresponds to what was previously defined.

There are two types of speed, linear and joint. If the linear speed is too high, there will be no time to properly orient the tool, which generates uncoordinated movements that may interfere with the proper functioning of the orientation

This may not be an issue while printing because the speed tends to be low. However, while simulating, it is necessary to take into account that using high speeds to accelerate the development and verification, may reproduce a false representation of the real movement.

6.3. Maximum Angle

During the testing phase, some situations were found where the orientation angles were too high, usually caused by the part's own geometry, this occurs when the closest point in the layer below is significantly distant in terms of X and Y coordinates.

On certain occasions, the print tool suffered rather abrupt or incorrect orientation variations (some points are out of the pattern).

This allows the user to set a maximum angle for the printing tool, which resolves this type of situation, preventing damage to the equipment and printing faults.

6.4. Configuration

As mentioned in **Chapter 2**, the configuration interferes with the angular position of the robot axes, setting their values within a certain limit.

The need to create a methodology applied to a large number of parts with different shapes is not possible with angular limitations as restricted as those applied by a given configuration.

Thus, the **ConfL\Off** instruction was applied, whose objective is to allow the robot to adapt to the best configuration for each movement.

6.5. G-code

When G-code is generated, some procedures, refered bellow, interfere with the proper functioning of the orientation algorithm.

6.5.1. Fifth point

So that the g-code instructs the robot to complete an entire perimeter lap, avoiding excessive deposition of material by not allowing the extruder to pass twice over the same point, it creates an extra point near the starting position. Once these points are too close, the program assigns them the same orientation, avoiding changes in the orientation for small displacements which cause abrupt movements.

The extra point was named the fifth point because while testing a quadrangular pyramid, the extra point corresponded to the fifth point of the layer, and a name for this type of point was not found.

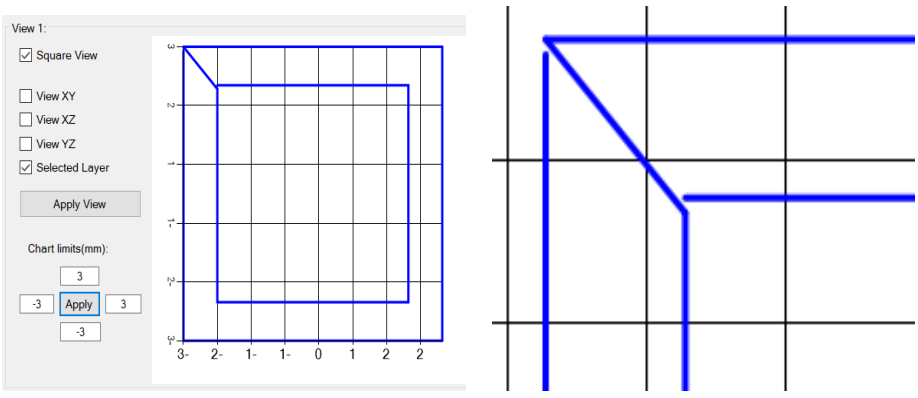

Figure 6.1 Fifth Point

6.5.2. Big variations for small offsets

The first approach of the algorithm was able to work correctly when the part to print did not include curved sections. In 3d modeling, the spheres are made of small lines.

When working with this type of part, the G code contains a large number of points, as a consequence, the number of orientations is equally large. By printing a circle with no defense against this enormous number of orientations, the TCP was constantly adapting its orientation. Besides, a minimum number of points may be out of the pattern, the TCP has to drastically change its orientation in short distances, leading to nonsense movements. To avoid this problem, a limiter for offset rotation was introduced, limiting the maximum degree variation between 2 consecutive points when they are within a defined offset. This limit and offset are defined by the user.

6.5.3. Several perimeters in the same layer

Slic3r has an option to choose the number of perimeter laps for the part. When executing the code for a part with more than 1 perimeter lap, certain orientations were calculated from points of different laps. When the algorithm searches for the nearest points on the layer below, it can pair points of different laps, which leads to incorrect orientations.

To solve this problem, a new variable has been introduced, which dictates the lap on which a given point is positioned. Using this new variable, the program was able to compare the first point of the lap with only the points of the first lap, correcting the errors. When this was introduced, a new error appeared. It occurs when the number of laps between 2 consecutive layers is different, as shown below.

To solve this last problem, when the number of laps in two consecutive layers change, the orientation is set to vertical. In red are represented the points that would be compared to calculate an orientation

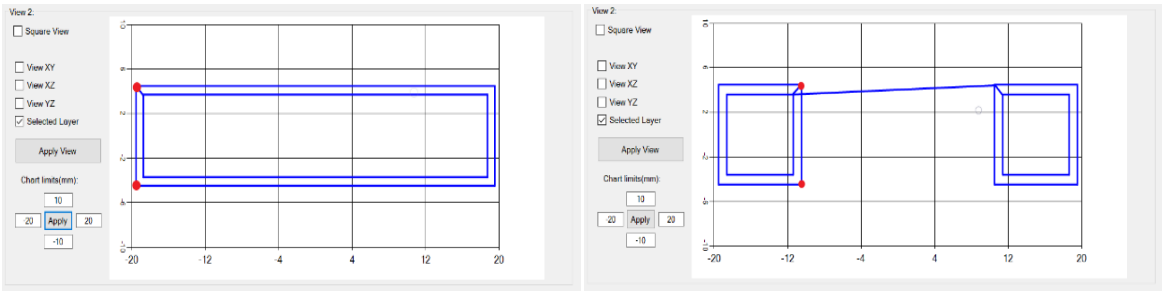

Figure 6.2 Perimeter Laps

6.6. Types of Orientation

As previously introduced, the rotation angles were calculated based on the nearest point in the layer below.

However, it is possible to establish several types of orientation by applying the method mentioned above.

6.6.1. Edge Orientation

During the visualization phase in Matlab, it was possible to identify that most of the points are on the edges of the part. Thus, applying the method defined in **Chapter 4**, the orientation corresponds to the line that defines the edge of the part.

In the case of parts with curved shapes, as there is no presence of edges, there will be no possibility to perform an orientation according to the edge of the part.

6.6.2. Two Point Average Orientation

This type of orientation does not directly apply the angles previously calculated by the orientation algorithm.

This Orientation was the main target of this dissertation. It orients the TCP in parallel to the corresponding face. To obtain this type of orientation, the program utilizes the average rotation angles from the departure and the arrival points to define the orientation between them.

6.6.3. Vertical Orientation

By default, the orientation of RobotStudio is vertical. It can be applied by setting all the rotations as null. This type of orientation is automatically applied as a solution to some problems described above

6.7. Types of Movement

The reason why different types of movement have been implemented has to do with the need to adapt the movement of the robot to the part to be manufactured. When two

different orientations are assigned in successive points, in RobotStudio, by default, the orientation of the robot is performed during the translation between two points, so as the TCP moves, it is reoriented according to the orientation of the next point.

When printing a geometric solid, for example, a pyramid. It must be intended that the orientation of the tool remains stable along the path between two points.

This way, a feature has been implemented that allows the user to choose which type of movement he prefers for each situation.

To achieve the different types of orientation it is enough to change the way the angles are implemented in the movement.

6.7.1. Movement with variable orientation

It allows the tool orientation to be variable along with the translation between 2 consecutive points.

If the orientation method is variable, the orientation has, at the starting point, the orientation of that same point. While the TCP moves to the next point, the orientation will gradually adjust to the orientation of the endpoint. When the TCP reaches the endpoint, it has exactly the orientation of the endpoint, allowing a smoother movement.

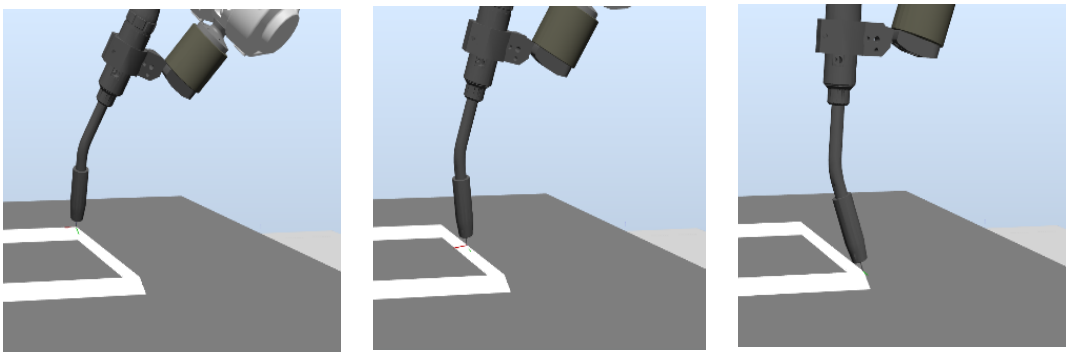

Figure 6.3 Variable Orientation

6.7.2. Movement with fixed orientation

If the chosen method is fixed orientation, the reorientation occurs before starting the translation movement, keeping the orientation stationary throughout the movement.

As the reorientation of the tool may take some time, for higher print speeds, the difference in reorientation speed and translation may be very different and consequently

create excess material. It allows maintaining a constant printing process along one side, not being indicated for parts with a very high concentration of points, since the reorientation requires a small stop on each point, turning the move less fluid.

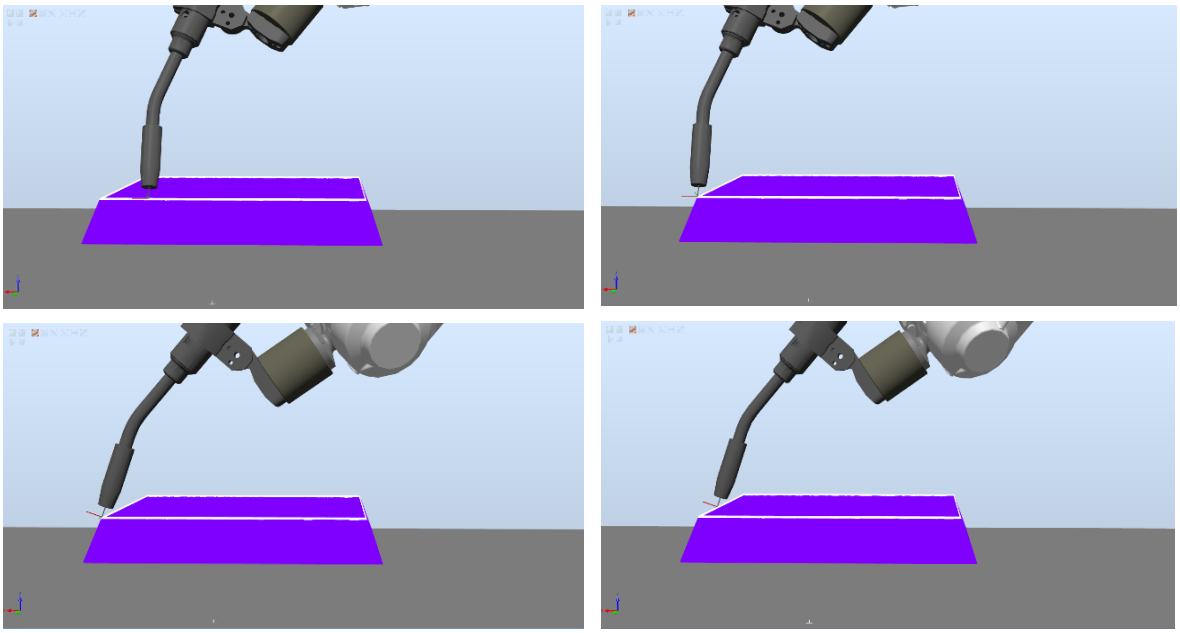

Figure 6.4 Fixed Orientation

7. DISCUSSION OF RESULTS

After all adjustments were completed, the final simulation was performed.

Two pieces were strategically chosen, to validate the method created. During the test phases, many pieces were tested. The most problematic geometries were undoubtedly, the ones containing circular paths. Thus, a piece containing spherical and planar geometries was drawn in **Figure 7.1.** The other piece is a hollow pyramid as it is easy to identify if the orientation algorithm is correctly working.

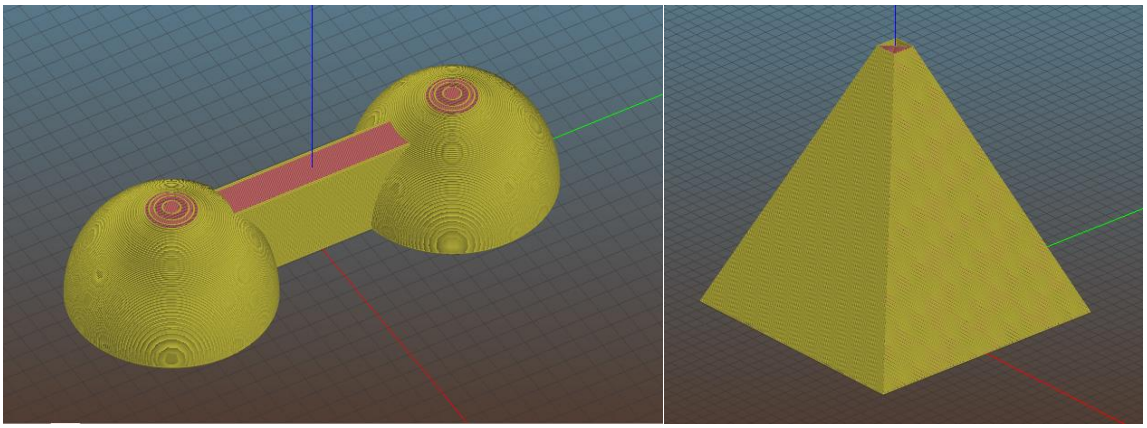

Figure 7.1 Final Test Pieces

7.1. About the types of movement to use on each situation.

The pyramid simulation shows that it should be more indicated to orient the printing tool before the translation. For a geometry like a pyramid, there are only 4 different orientations per path (one for each face), which is a considerably low number of orientation changes. As the points are significantly distant, there is no need to smooth the robot movements, so orienting the printing tool before the translation to the next point maintains the orientation constant along the path.

On the other hand, for the spherical piece orienting before the movement does not seem a good approach. Circular paths are composed of innumerous points. This way, the printing tool orientation is constantly changing, which is a problem, since this type of movement requires a small portion of time to readapt to the orientation of the next point. This causes an inconstant movement, certainly not appliable for this piece. To resolve this situation, the orientation during movement was applied. With this method, the printing tool movement became consistent, smooth and the orientation was visually correct.

7.2. Types of orientation

Both pieces were tested with edge and two-point average orientation.

The two-point average orientation seemed to be correct and appliable for both situations, as the printing tool kept its orientation parallel to the surface.

The edge orientation, although the movements were possible for both pieces, it leaves some uncertainty about its application.

7.3. Variation of the orientation

As mentioned before, the control of the orientation by offset, solved the inconsistent pattern of points placement. However, the user must be aware that the distance between successive points is crucial to define the correct offset to choose and the corresponding maximum rotation to be applied. For the pyramid piece, this feature is not relevant as its points are distant enough.

8. CONCLUSIONS

This thesis presents a solution that aims to implement a methodology to adapt the orientation of the printing tool connected to an industrial robot. The main objective of this solution would be to improve the printing process of metallic components, by reducing the printing time and the material needed. For this purpose, an algorithm was developed, based on the geometry of the part, implementing an orientation parallel to its surface.

All the tests were carried out on the RobotStudio interface. Although there are studies that demonstrate the benefits of orienting a print tool parallel to the surface to be printed, it is important to perform tests on the real system to ensure the correct functioning of the system and verify the properties of the printed part.

Throughout the development of the program, it was necessary to implement several methods of orientation and movement of the robot to ensure the adaptation for different part geometries. Visually, circular and straight paths must be executed with different types of movement. However, for confirmation purposes, real tests must be done.

Although the methodology has worked correctly for the two parts studied in more detail, it is necessary to perform more tests.

In conclusion, although all the objectives have been achieved, the program is not fully automatic. It cannot automatically identify the best methodology or combination of methodologies to apply to each part, which forces the user to intervene in the selection of methods to apply.

8.1. Future Work

This solution should be interpreted as an initial approach that opens up possibilities for the following developments and/or improvements:

- Testing in the real system the applicability of the orientation adjustment during printing
- Study the mechanical behavior of the printed parts
- Complement the program with an adapted slicing system
- Automate the selection of the best method

BIBLIOGRAPHY

ABB. 2018. “Technical Reference Manual, RAPID Instructions, Functions and Data Types OLD.”

“An Introduction to Wire Arc Additive Manufacturing [2020 Update] - AMFG.” https://amfg.ai/2018/05/17/an-introduction-to-wire-arc-additive-manufacturing (October 21, 2020).

Champlain, John. 2015. “Understanding Gcode Commands.” (February). http://www.picengrave.com/Pic Programs Page/PDF Files/misc/Understanding Gcode.pdf.

DebRoy, T. et al. 2018. “Additive Manufacturing of Metallic Components – Process, Structure and Properties.” *Progress in Materials Science* 92: 112–224.

Diebel, James. 2006. “Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors.” *Matrix* 58: 1–35. ftp://sbai2009.ene.unb.br/Projects/GPS-IMU/George/arquivos/Bibliografia/79.pdf.

Frazier, William E. 2014. “Metal Additive Manufacturing : A Review.” 23(June): 1917–28.

IDC. “IDC’s Worldwide Semiannual 3D Printing Spending Guide.” https://www.idc.com/getdoc.jsp?containerId=prUS44619519 (August 4, 2020).

“IEDI.” https://www.iedi.org.br/cartas/carta_iedi_n_899.html (October 26, 2020).

“Inventor | Mechanical Design & 3D CAD Software | Autodesk.” https://www.autodesk.com/products/inventor/overview?plc=INVPROSA&term=1-YEAR&support=ADVANCED&quantity=1 (April 22, 2020).

Jiang, Jingchao, Jonathan Stringer, Xun Xu, and Ray Y. Zhong. 2018. “Investigation of Printable Threshold Overhang Angle in Extrusion-Based Additive Manufacturing for Reducing Support Waste.” *International Journal of Computer Integrated Manufacturing* 31(10): 961–69. https://doi.org/10.1080/0951192X.2018.1466398.

Kasinski, Abraham, Carlos Pires, Oliveira Dias, and Pedro Eberhardt. 2005. “Instituto de Estudos Para o Desenvolvimento Industrial D ESINDUSTRIALIZAÇÃO NO B RASIL ?” : 1–23.

Kazanas, Panagiotis et al. 2012. "Fabrication of Geometrical Features Using Wire and Arc Additive Manufacture." *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 226(6): 1042–51.

"MATLAB." https://www.mathworks.com/products/matlab.html (May 15, 2020).

Mellor, Stephen, Liang Hao, and David Zhang. 2014. "Additive Manufacturing: A Framework for Implementation." *International Journal of Production Economics* 149: 194–201.

Mohan Pandey, Pulak, N. Venkata Reddy, and Sanjay G. Dhande. 2003. "Slicing Procedures in Layered Manufacturing: A Review." *Rapid Prototyping Journal* 9(5): 274–88.

"MX3D Bridge | MX3D." https://mx3d.com/projects/mx3d-bridge/ (October 27, 2020).

Pires, J.Norberto. 2020. *Robótica (e Biónica)*.

"RobotStudio." https://new.abb.com/products/robotics/robotstudio (April 27, 2020).

"Slic3r." https://slic3r.org/ (April 22, 2020).

"SOFTWARE | MX3D." https://mx3d.com/metalxl/ (June 12, 2020).

"TAKENAKA CONNECTOR | MX3D." https://mx3d.com/projects/takenaka-connector/ (June 12, 2020).

"Titanium Pressure Vessel for Space Exploration Made Using 3D Printing." https://www.pesmedia.com/titanium-pressure-vessel-space-exploration-additive-manufacturing/ (May 19, 2020).

Wu, Bintao et al. 2018. "A Review of the Wire Arc Additive Manufacturing of Metals: Properties, Defects and Quality Improvement." *Journal of Manufacturing Processes* 35(August): 127–39. https://doi.org/10.1016/j.jmapro.2018.08.001.