



Fakultät für Luftfahrt, Raumfahrt
und Geodäsie

Extrusion Based Additive Manufacturing of Large-Scale Lattice Structures

Mateo Guarch, María

Wissenschaftliche Arbeit zur Erlangung des Grades B.Sc.
an der Fakultät für Luftfahrt, Raumfahrt und Geodäsie der Technis-
chen Universität München.

Betreut von Univ.-Prof. Dr.-Ing. Klaus Drechsler
Lehrstuhl für Carbon Composites

Wissenschaftlicher Mitarbeiter Consul, Patrick, M.Sc.
Lehrstuhl für Carbon Composites

Eingereicht von Maria Mateo Guarch
Adolf Kolping Straße 1
80336 München
Tel.: +34 674 631 224
Matr.-Nr.: 03727397

Eingereicht am München, den 21.09.2020

Technische Universität München
Fakultät für Luftfahrt, Raumfahrt und Geodäsie
Lehrstuhl für Carbon Composites
Boltzmannstraße 15
D-85748 Garching bei München

Tel.: +49 (0) 89 / 289 – 15092
Fax: +49 (0) 89 / 289 – 15097
Email: info@lcc.mw.tum.de
Web: www.lcc.mw.tum.de

Bachelor's Thesis

Extrusion Based Additive Manufacturing of Large-Scale Lattice Structures

Additive manufacturing by Material Extrusion, also known as 3D printing, uses thermoplastic polymers to create complex structures layer by layer. Extrusion-based processes allow fiber reinforcement orientation. By an appropriate choice of polymer and fiber content, 3D printed structures can be tailored to specific material properties. This makes them suitable for a variety of applications such as lattice structures composed of a network of nodes and beams or struts. These structures are lightweight while achieving high strength using minimal material in a large volume. The mechanical performance of carbon fiber reinforcement can be optimally exploited in beam and strut like structures if they are oriented along the strut.

This bachelor thesis will focus on the process development to print large scale lattice structures with a single screw extruder guided by a six-axis robot. The work will include machine development by analyzing the cooling requirements to enable free-form material extrusion, material selection based on a short market research, lattice cell development and generation and finally the demonstration of the thesis work by printing a large-scale lattice structure. The main work will be on the lattice cell development and generation to ensure collision free manufacturability on the one hand and adaptability and mechanical performance on the other.

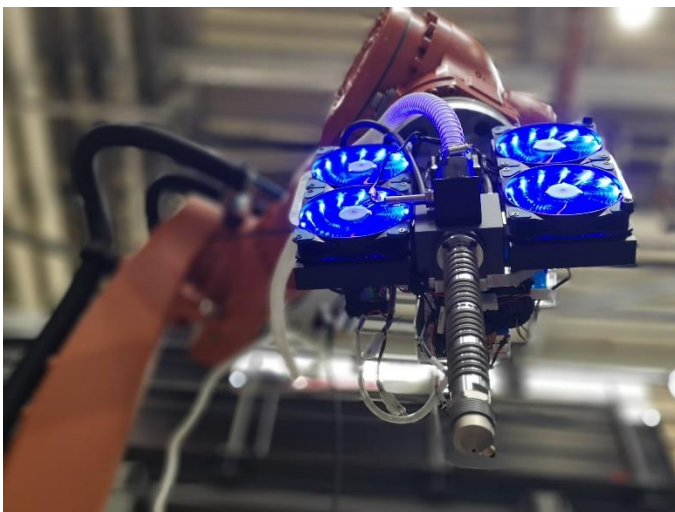


Figure: Printer used for validation

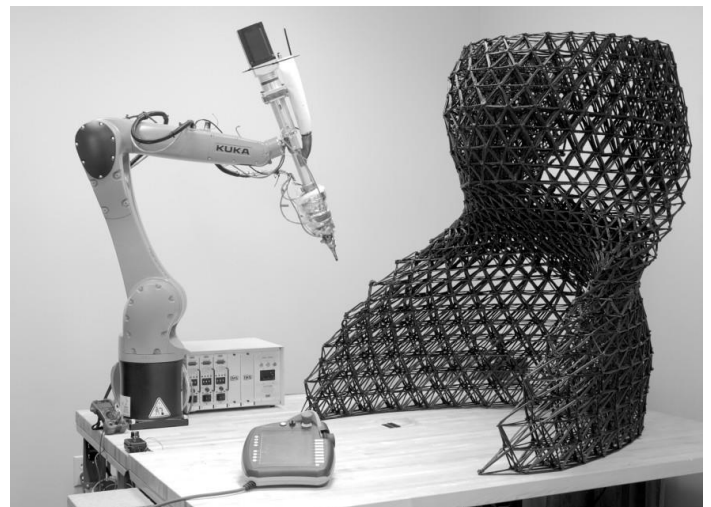


Figure: Example of large-scale lattice structures for architecture

Research focus of the thesis

- Cooling concepts for free-form material extrusion
- Material selection
- Lattice cell development and generation
- Validation and demonstration on robot-based 3D printer

Requirements

- Basic understanding of thermoplastic additive manufacturing
- Basic understanding of thermoplastic composites
- Interest in automated production

Starting date: Now

For more details please contact:

Patrick Consul, Room 1426, FSZ, Tel. +49 89 / 289 – 15101, Patrick.Consul@tum.de

Ehrenwörtliche Erklärung

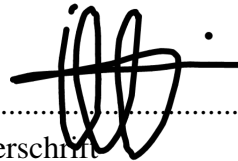
Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht.

Die Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

München, 21/09/2020

.....

Ort, Datum

A handwritten signature in black ink, consisting of several loops and a horizontal line at the end.

.....

Unterschrift

Übersicht

Der 3D-Druck ist eine schnell wachsende Technologie. Jeden Tag entstehen neue Anwendungen und Möglichkeiten. Gegenwärtig basieren die meisten additiven Fertigungstechniken auf der Abscheidung aufeinanderfolgender Materialschichten, die zur Schaffung eines festen Objekts führen. Die Technologie ändert sich ständig, und es ist eine neue Art und Weise entstanden, 3D-Formen zu bauen, die während der gesamten Arbeit der Dissertation vorgestellt wird: eine neue Druckmethode, die die gleiche Ausrüstung und das gleiche Material wie bisher verwendet und in der Lage ist, selbsttragende Gitterstrukturen zu erzeugen, weit entfernt von den Zwängen der Modellierung durch Schmelzabscheidung.

Die Möglichkeit, solche Strukturen drucken zu können, eröffnet ein neues Feld von Möglichkeiten, die Produktionsketten zu beschleunigen. Im Ergebnis wird die verarbeitende Industrie effizienter sein und weniger Abfall erzeugen, was einer nachhaltigeren und kosteneffizienteren Zukunft entgegengeht.

Dieses Projekt stellt einen möglichen Workflow zum Drucken von Gitterstrukturen vor. Von der Entwicklung und Montage eines geeigneten Kühlkonzepts bis hin zur Auswahl des richtigen Materials und der Erstellung des Codes. Der erzeugte Code wird zur Herstellung der Gitterstrukturen in einem 6-Achsen-Roboterarm eingesetzt, um die Methodik zu validieren.

Abstract

3D printing is a rapidly growing technology. Each day new applications and opportunities arise. Currently, most Additive manufacturing techniques are based on the deposition of consecutive layers of material, that results in the creation of a solid object. Technology is constantly changing and a new way to build 3D forms has arisen, which is presented throughout the work of the thesis: a new printing method that uses the same equipment and material used until now and is able to create self-standing lattice structures, far away from the constraints of fused deposition modelling.

Being able to print such structures opens a new field of opportunities, that could speed up production chains. As a result, the manufacturing industry will be more efficient and less waste will be generated, heading towards a more sustainable and cost-efficient future.

This project presents a potential workflow to print lattice structures. From the development and assembly of a proper cooling concept to the selection of the right material and the creation of the code. The created code will be applied to manufacture the lattice structures in a 6 axis robotic arm to validate the methodology.

Contents

Ehrenwörtliche Erklärung	IV
Aufgabenstellung	IV
Übersicht	V
Abstract	VI
Table of contents	VII
Nomenclature	X
List of abbreviations	XI
1 Introduction	1
1.1 Goal of the thesis	1
1.2 Outline	1
2 Principles	3
2.1 Additive Manufacturing	3
2.2 G-Code	3
2.3 Large scale manufacturing	4
2.4 Free form printing	5
2.5 Related work	7
2.5.1 MotoMaker	7
2.5.2 Free-hanging 3D printing method	8
3 Cooling concept for freeform printing	9
3.1 Why is cooling necessary?	9
3.2 Design of the cooling system	9
3.3 Assembly	11
4 Material selection	16
4.1 Thermoplastics	16
4.1.1 ABS	17
4.1.2 Polycarbonate	17
4.1.3 Polypropylene	17
4.2 Final Selection	18

5	Lattice structures	19
5.1	Advantages of lattice structures	19
5.2	Applications	19
5.2.1	Aerospace	20
5.2.2	Architecture	20
5.3	Generation of the lattice structures	21
5.3.1	Simple cube design	22
5.3.2	Horizontal panel design	25
5.3.3	Plane orientation design	32
6	Validation	43
6.1	Simple cube	44
6.1.1	Cube N°1	44
6.1.2	Cube N°2	46
6.1.3	Cube N°3	48
6.1.4	Cube N°4	49
6.1.5	Cube N°5	51
6.1.6	Cube N°6	53
6.1.7	Cube N°7	55
6.1.8	Cube N°8	57
6.1.9	Cube N°9	59
6.1.10	Cube N°10	61
6.1.11	Cube N°11	63
6.2	Horizontal lattice structure	65
6.2.1	Horizontal panel N°1	65
6.2.2	Horizontal panel N°2	69
6.3	Orientation change	73
6.3.1	Pentagon N°1	73
6.3.2	Pentagon N°2	75
6.3.3	Pentagon N°3	77
7	Future improvements	80
7.1	Cooling system	80
7.2	Bed adhesion	81
7.3	Material	81
7.4	Grasshopper	82
8	Conclusions	83
	Bibliography	84

List of figures	88
List of tables	89
Annex	90
CD	90

Nomenclature

Formelzeichen	Einheit	Beschreibung
<i>E</i>	[mm]	Extrusion
<i>d</i>	[mm]	segment length

List of abbreviations

Abbreviation	Meaning
ABS	Acrylnitril-Butadien-Styrol
CAM	Computer Aided Manufacturing
CFRP	Carbon fiber reinforced plastic
CFRTP	Carbon fiber reinforced thermo plastic
FDM	Fused Deposition Modeling
AM	Additive Manufacturing
FFF	Fused Filament Fabrication
LFAM	Large format Additive Manufacturing
PEEK	Polyetheretherketone
PET	Polyethylene Terephthalate
PP	Polypropylene
PA	Nylon
PC	Polycarbonate
PETG	Polyethylene Terephthalate Glycol
PRC	Parametric Robot Control
KRL	Kuka Robot Language

1 Introduction

Additive manufacturing (AM) also known as 3D printing, is the creation of a three-dimensional object from a CAD model. It was first introduced in the 80s as a new and exciting approach towards the fabrication and automation. Materials such as thermoplastic polymers are used to create layer by layer the desired product. If the material is chosen correctly, its properties can be known and used for specific purposes.

There exist several types of 3D printing techniques, Fused Deposition Modeling (FDM) is a well known traditional procedure where the object is built by stacking planar layers onto each other with the help of support material, it is basically printing in a 2D manner, depositing material layer by layer. Each layer is built by extruding melted material, normally thermoplastic polymers through the nozzle. The method exposed in this thesis goes a step forward, it targets points in 3D space instead of layering. With the help of a six axis robot we are able to step out of the typical FDM printing creating fully 3D such as lattice structures.

Lattice structures are topologically ordered, three dimensional open-celled structures composed of one or more repeating unit cells. These cells are made of struts and connected through specific nodes. By controlling the parameters of these structures one can achieve unique properties which can be used in a variety of engineering fields [9], for example in aerospace application, light weight structures can be produced, with high strength-to weight ratio properties.

1.1 Goal of the thesis

The goal of this work is to investigate the process and development of large lattice structures. Designing and manufacturing the cooling requirements needed to generate this type of structure, researching the right materials for the printing and generating a lattice structure. Finally the work will be validated by printing such a structure with a single screw extruder guided by a six-axis robot.

1.2 Outline

The thesis has been divided into different sections to obtain the best result possible. Firstly, the principles of 3D printing will be explained, some important references and related work are also mentioned. Next, the cooling system for the extruder will be designed and assembled. Then, the material selected and the reasons behind this selection

will be explained. In the next chapter, the workflow followed for the code generation will be commented and finally we will validate the workflow in the KUKA robotic arm. A chapter containing future improvements and the conclusions has also been added to sum up the work done.

2 Principles

This chapter gives an insight of the basic principles that should be known about 3D printing and specifically 3D printing large structures, lattice structures which is the focus of this thesis. First a short and general overview of the 3D printing process is given, then we focus on larger structures and finally we discuss the method we will be using throughout the work, freeform 3D printing, a couple of examples are mentioned to give a clear picture of the objective of this work.

2.1 Additive Manufacturing

Additive Manufacturing was first introduced in 1984 by Chuck Hull who patented a stereolithography fabrication system which consisted in layers being added by curing photopolymers with ultraviolet light lasers.

Since then AM has been of great interest to researchers and industries for its ability to create objects from scratch from a range of different materials such as plastics, ceramics and even metals. Moreover, AM technology encloses both minimal waste and time efficiency, as products are quickly designed and manufactured without hardly wasting any material, that could be reused in future prints.

The 3D printing process can be separated in different steps. Firstly, the model is generated through Computer Aided Design (CAD), which is saved as an STL file which describes the geometry of the model. The next step is to slice the model, 3D printers are not able to print the STL model directly, it has to be translated into a language that can be read by the printer, which is called G-code. Finally, the piece has to be printed, thanks to the G-code the printer gets the instructions and executes them in a fully automated manner.

The products printed with this technology are normally of a modest size, limited by the size of the print bed and the distance between the bed and the extruder. Trying to print larger and bigger models is something worth looking into.

2.2 G-Code

Since we are going to be working with 3D printers, it is important to know how they work and what type of computer language they use to make the necessary modifications if necessary.

G-Code is a control language used in Computer Aided Manufacturing (CAM), in manufacturing processes such as milling, drilling or 3D printing. This programming language

tells the machine how to move from the necessary toolpaths, to the speed of the spindle, it is even capable of controlling the heat via temperature control. Each line of code is responsible for one particular action, 3D printing a part could be composed of hundreds of lines of code that work together to create the desired part.

Some G-Code commands that are frequently used in 3D printing are:

M83: Set extruder to relative mode, makes the extruder interpret extrusion values as relative positions.

G90: use absolute coordinates, the position is defined with reference to part zero.

G21: Programming in millimeters (mm).

G92 E0: Set position, no extrusion.

M104 S280: Set the hotend temperature to 280. The **M** stands for miscellaneous functions.

M109 S280: Wait for the hotend temperature to reach 280.

G1 X200 Y200 Z2 F1200 E1: Linear movement to $x,y,z=200,200,2$ with a feedrate of 1200 mm/min, it moves the extruder to 1 mm. The **G1** command is the most used command in 3D printing since the building of the object depends on it.

M84: Stop idle, disable the motor.

2.3 Large scale manufacturing

Large-scale AM machines are relatively new but they offer countless opportunities, they are able to construct objects of nearly unlimited geometry in a fast way and very automated.

Large format Additive Manufacturing (LFAM) is a large scale-polymer extrusion based on AM technique. LFAM is similar to FFF in that a molten thermoplastic is extruded along a tool path to generate a 3D part. Instead of using a heated chamber to melt a thermoplastic filament, LFAM uses a single screw extruder to melt pelletized feedstock. The single screw extruder increases the deposition rate up to 50 kg per hour, which is about 200 times faster than conventional AM machines [1]. This means that large structures can be fabricated faster and at a lower cost. LFAM applications and opportunities are expanding rapidly.

Another way to achieve LFAM is to use multiple printers, a multi-robot 3D printing system [14]. It consists of a host computer, a communication interface and four robotic arms with end effectors used for the extrusion of the material. The main computer sends the commands to the robot controllers and they make the precise movements and extrusion required. The procedure is shown in Figure 2.1. Each robot would get single

instructions of particular tasks in a fixed position, the whole system creates a collaborative 3D printing environment. It is important to locate each robot in a particular position, making sure that one robot does not overstep another.

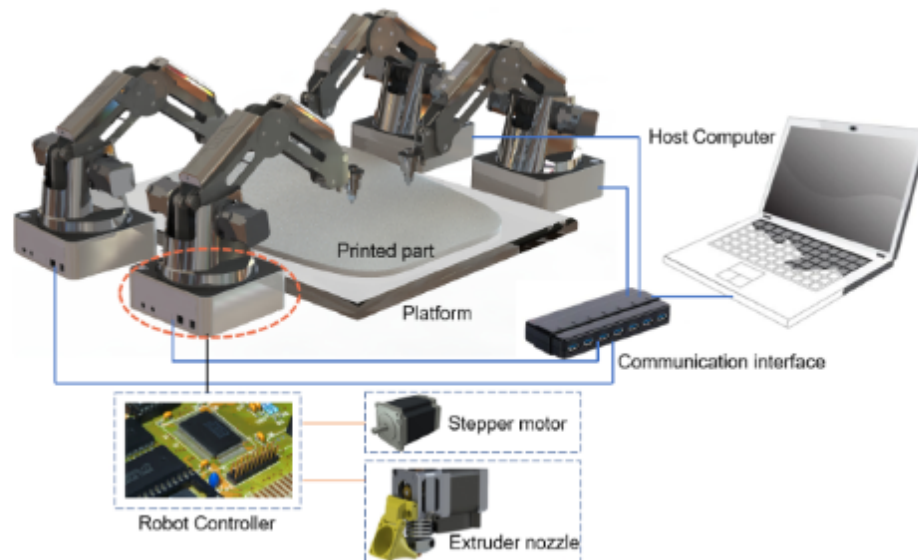


Figure 2.1 : Multi-robot 3D printing system [14]

2.4 Free form printing

Free form printing comes from fused filament fabrication (FFF). FFF is one of the most used 3D printing techniques, where the nozzle extrudes the material along a planar path, each layer fuses with the previous, forming the desired model. In a similar way Free Form printing extrudes the material along a desired path but the material solidifies immediately after being extruded, which means that more complicated structures can be attempted to print.

A Mediated Matter Group in MIT Media Lab did a project called "Freeform 3D printing: Towards a Sustainable Approach to Additive Manufacturing" [11] and tackles a novel design approach for 3D printing which leaves behind the additional auxiliary structures previously needed to print. Here a 6-axis KUKA robotic arm is used as the 3D platform onto which a thermoplastic extruder is attached seen in Figure 2.2. With the elimination of the support structures, the movements of the nozzle can be more complex, also less material is wasted and the model can be generated at a faster rate. Overall, this shows a more sustainable approach of manufacturing products.

This new and exciting way of 3D printing has also been developed and researched by other companies such as Branch Technology and AiBuild. Branch Technology is a company which applies freeform 3D printing to the design and construction of architectural

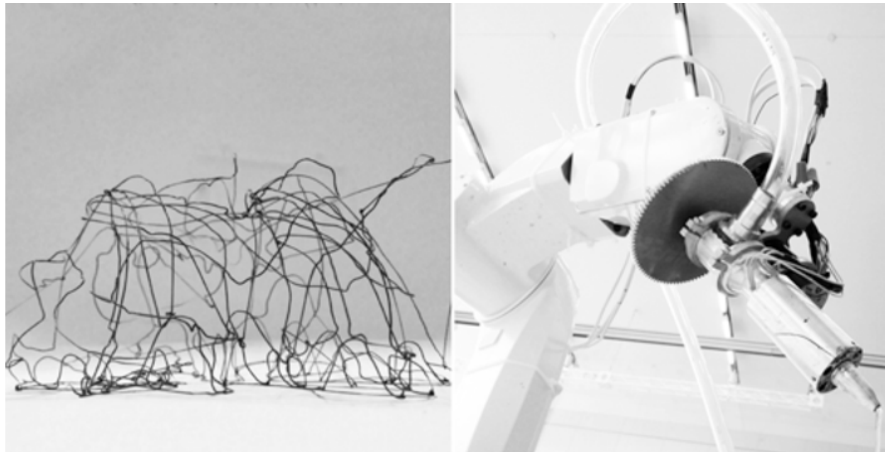


Figure 2.2 : ABS Filament freeform extrusion [11].

structures. They have developed their own Cellular Fabrication Technology in order to achieve those strong and efficient structures. This company is an important precedent of this thesis as it makes what we want to achieve. As seen in Figure 2.3 a mesh made up of unit cells, lattices is made to form the 3D printed structure. The empty space in between the lattices is later filled with a secondary support, concrete to give it more strength. The combination of both, the lattice and the concrete creates a structure that is able to withstand large loads and can be used in construction.

Ai Build [5] is a company focused in Artificial Intelligence and Robotic technologies for large scale additive manufacturing. Their goal is to develop their own software of tool-path generation and optimisation. It uses freeform to print structures without material support, dividing large geometries into cells, or lattices, which make up a continuous path to be printed on.



Figure 2.3 : Branch Technology [2]



Figure 2.4 : AiBuild [5]

2.5 Related work

Several researchers have already attempted to use free form printing to fabricate lattice structures. In the previous section two big companies have been mentioned: Branch Technology and AiBuild that are the main driving motors of this new and exciting technology. Nevertheless, there are smaller projects that are strongly related with the outline of this project and have been of great help and inspiration for the development of the work.

2.5.1 MotoMaker

A joint collaboration between the University of Malaysia, Florida Institute of Technology and the South Dakota School of Mines and Technology have developed a robot called Motomaker [6], a robot of FDM for multiplane and 3D lattice structure printing. The robotic arm has six degrees of freedom and a fused deposition modeling printing system. The extruder is composed of the extruder head and a cooling system to heat down the filament when being extruder so it solidifies quickly to achieve this they use a cooling fan and compressed air. The addition of a fan to a system is very straight forward and could be a good option for the cooling system we need.

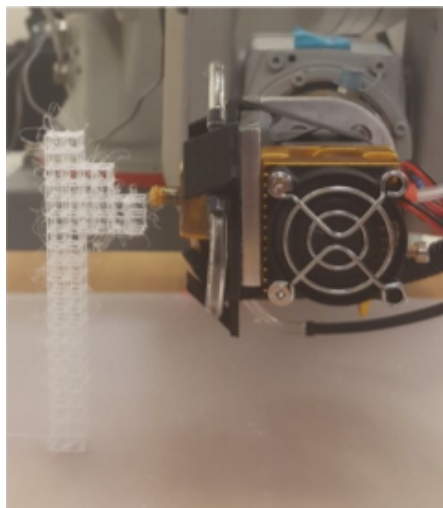


Figure 2.5 : MotoMaker [6]

In Figure 2.5, the procedure of printing with the Motomaker robot can be seen. Similarly to our case, they are trying to print a lattice structure, the programming language that they are using for the construction of the lattice structure is Matlab. The printing process very similar to the one we will use. The toolpath (G-Code) is stored in a interface program and sent to the robot controller via ethernet, the controller tells the robotic arm when to move and how fast. At the same time the interface program connects with the

extruder controller and feeds the corresponding operations to the extruder feed motion, the heating element and the cooling fan.

2.5.2 Free-hanging 3D printing method

The College of Mechanical and Electrical Engineering of Nanjing University of Aeronautics and Astronautics released a paper [8] in which they present a novel free-hanging 3D printing method for continuous reinforced thermoplastic lattice truss core structures. They propose carbon fiber reinforced thermoplastic (CFRTP) as the printing material for the lattice structures since it has low density, high strength and it can be recycled which due to increasing regulations it is a great advantage. They used PLA as the reinforced thermoplastic and analyse the structural improvements that are given with this method.

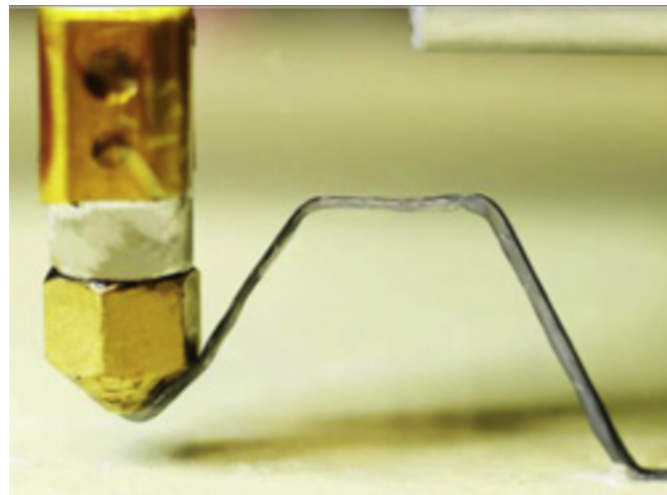


Figure 2.6 : Free hanging 3D printing [8]

The workflow followed for this method is different than the MotoMaker method [6]. This method is more focused on the design requirement, the material that will be used, the fiber, matrix, temperature for printing... and the geometry of the lattice structure, the configuration, relative density and lattice topology. Once this has been sorted out, the toolpath is generated and the G-Code is sent to the free hanging 3D printer. Figure 2.6 displays the printing procedure of a lattice structure. Similarly to the previous case, this method also uses cooling flow to quickly cool down the filament that is being printed. They have implemented two pipes that push air down to the nozzle where the filament is being extruded. This could be a good idea for the cooling system that we have to design.

3 Cooling concept for freeform printing

This chapter gives an insight of how the cooling system for the freeform printing was design, from why is it important to have such devices to cool off the printed structure to the actual design and manufacture and implementation of the system designed.

3.1 Why is cooling necessary?

When 3D printing the material moves through a hot end, here the material is fully controlled by the G-Code. However, when the material leaves the nozzle it cannot be controlled by the code anymore. For this reason, it is of out most importance to make sure that the material sticks to its required position. In order to achieve this a cooling system should be installed.

Cooling it is extremely important for freeform printing, since we rely on the position of the previously printed cell unit to stick the following cell unit. The material chosen as filament is key since it has to cool down very fast, but a cooling aid could also be beneficial.

Once this has been clarified we can start looking at what type of cooling system would be best suited. There are several types of cooling systems used in 3D printing, ranging from water cooling, air pumps to fans. The last ones are the most widely used since they can be easily controlled by the G-Code.

3.2 Design of the cooling system

Before starting to design the cooling system, the extruder and robot arm should be analysed. As it has been said before, the extruder is attached to a six axis robotic arm so the system designed has to be able to withstand the movements of the arm throughout the six axis. The extruder that we will be working with can be seen in Figure 3.1.

At first glance, it would make sense to place the cooling system near the extruder, however, since the bottom part of the extruder will be extremely hot due to melted material coming through, the cooling system should be placed elsewhere in order to avoid the melting of the system. Perhaps, on the top part of the extruder and then redirect the air flow to the bottom to cool the lattice structure that it is being printed.

There exist several types of cooling options, but fans are probably the most used ones. Placing one or two fans at the top of the extruder and then redirecting the flow downwards is the best option. In order to get the flow at the bottom of the structure a silicone tube can be attached to the fan and then with some kind of pipe push the air outwards. However, one big question arises. Does the air pushed by the fans have enough power to travel down a pipe and finally cool down the printed part?



Figure 3.1 : Pulsar Extruder.

There are two main types of fans: axial fans or radial fans. Radial fans are those in which the air inlet and outlet goes through perpendicularly, the air is sucked in through the central area and it goes out through a duct. This type of fan is very useful to cool specific areas. The air outlet goes through a small area causing the velocity and the pressure of the air to increase.

Therefore, radial fans will be used as the main power source to cool the printed structure. The characteristics of the fan selected can be seen in Table 3.1. The initial idea is to install a couple of these fans on the top of the robot and then make some kind of structure to redirect the flow from the duct to the bottom of the extruder to be able to cool the part.

	Radial Fan
Name	WINSINN 5015
Nominal voltage	24 V
Nominal current	0.1 A
Power	2.4 W
Noise	27dba
Dimensions	50x50x15 mm

Table 3.1 : Data of the Radial Fan

CATIA was used to design the structure that will support the fans and attach them to the robot. The final part designed can be seen in Figure 3.2. It can be seen how it can

support up to two fans which are screwed into the structure. Also the main structure is assembled into the robot thanks to two screws at the top. The air coming out of the fans is redirected to the bottom of thanks to a transition duct from a rectangular form to a circular one. A silicone tube will be later attached to the circular duct to redirect the flow of air to the pipes at the bottom of the extruder.

The bottom structure is shown in Figure 3.3. It will be attached to the bottom of the pulsar extruder. As it can be seen it has two holes so the copper pipe can pass through it. To make sure that it does not dance around and move while printing, to the sides of the plate small holes have been done so screws can be placed to limit the overall movement. The copper pipes will go down until the bottom of the extruder and it will direct the flow of air to the desired printed structure.

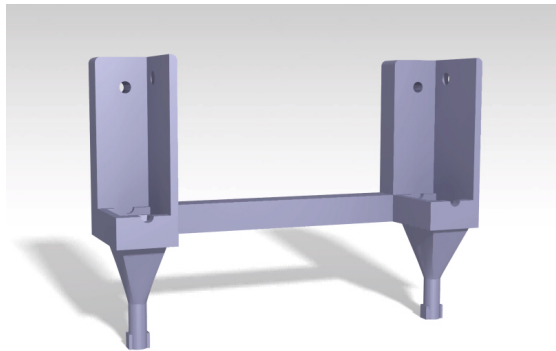


Figure 3.2 : Design of the top structure

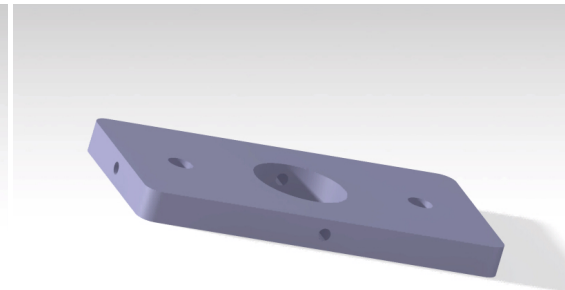


Figure 3.3 : Design of the bottom structure

3.3 Assembly

Once the design of the cooling system has been done, the assembly and manufacturing can be done. The top support structure seen in Figure 3.2 was 3D printed in one of the 3D printers of the institute. The bottom structure seen in Figure 3.3 was water jet cut, the base plate is an aluminium plate, extremely malleable so we could later make the small screw holes without any problem. While assembling the system we encountered some problems. To start of, the big hole of the bottom structure was too small to fit through the extruder so we had to fine it down and polish it until it fitted. For the pipes we used copper tubing which allowed us to twist and bend the pipes as needed. The dimensions of the tubes were 8 mm outer diameter and 6 mm inner diameter. For the silicone tubing that will go from the fans to the copper tubes, 7 mm inner diameter and 12.6 outer diameter was used. Therefore, the silicone tubing could be pulled over the copper tubing and make an air tight connection. All the screws used in both parts were M4 screws.

The system assembled can be seen in Figure 3.6. It can be seen how the system is mounted into the pulsar extruder. The top structure was placed at the rear part of the

extruder, where the rest of the cooling system was, what we achieve by doing this is having all the cables at the rear part of the robot and therefore, we avoid future cabling problems and collisions of the robotic arm with the cables.

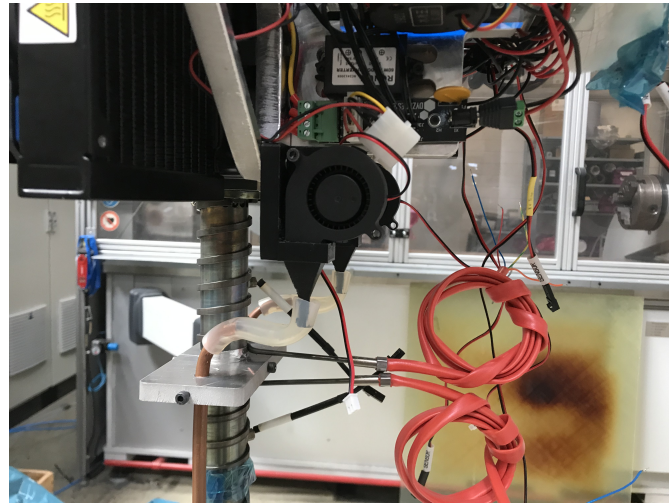


Figure 3.4 : Side view of the top structure

The top structure can be seen in more detail in Figures 3.4 and 3.5. Here it is easy to see how the fans are finally mounted onto the structure and how the silicone pipe goes from the circular duct of the top structure to the copper tubing. We tried to make the path from the top structure to the copper tubes as small as possible, in other words, to make the silicone tube very short so the flow has to travel very little space before going into the copper tubes and down to the bottom part of the extruder.

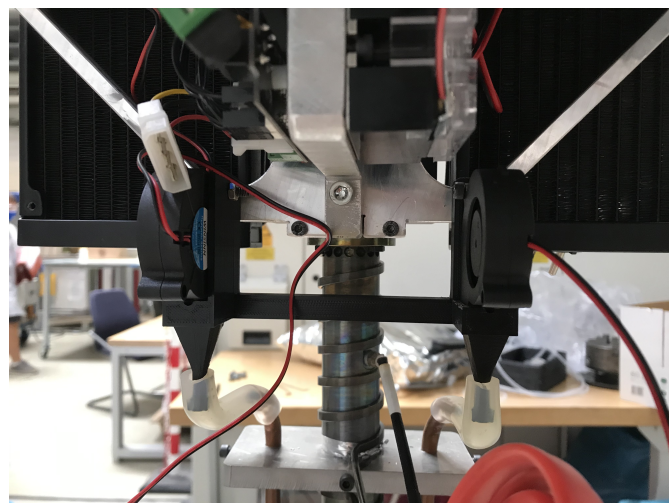


Figure 3.5 : Rear view of the top structure

In Figure 3.7 a closer shot of the bottom structure is displayed. Here, it can be seen how the copper pipes are placed so the flow exhausted goes out directly into the nozzle and cools down the filament that is being printed. Also it is clearly seen how the silicone

tubes are attached to the copper tubing, and how this is placed in the bottom structure, tightened with screws so it does not move when tilting the extruder when printing.



Figure 3.6 : Front view of the system

Once the tubes were installed we proceeded to try the system by turning on the fans. The first impression was that the fans were hardly exhausting any air. Then we realised that instead of pushing air inwards the fans were pulling the air outwards. After correcting this we tried again and we saw how the fans were indeed working in the right direction but the amount of air that it was exhausted was much smaller than we expected. We dismantled the fans once again to see how much air they were pushing without the structure and we saw that it was quite big. Therefore, the problem was the support structure and the pipes. Probably, throughout the silicone tube and in the top support structure some air was lost. The first solution we approached was to seal all the edges that could be exhausting air with an special duct that is almost like vacuum. With this, we made a big improvement, but not good enough.



Figure 3.7 : Closer shot at the bottom of the extruder

Next up, we unscrewed the the top structure that was attached to the support of the extruder and we rotated it 90°, as it can be seen in Figure 3.8. By doing this we redirected the flow of the fans, making it easier to flow since it no longer has to bend to get into the cooper tubes. Now the silicone tube is straight and the flow goes in directly. This made a big improvement on the overall flow of the exhaust air. Still, it was less air than expected but for the material that we were going to use to print it was good enough. Nevertheless, for future improvements, a fan with more power, or redesigning the silicone-copper tube system would be a good idea.

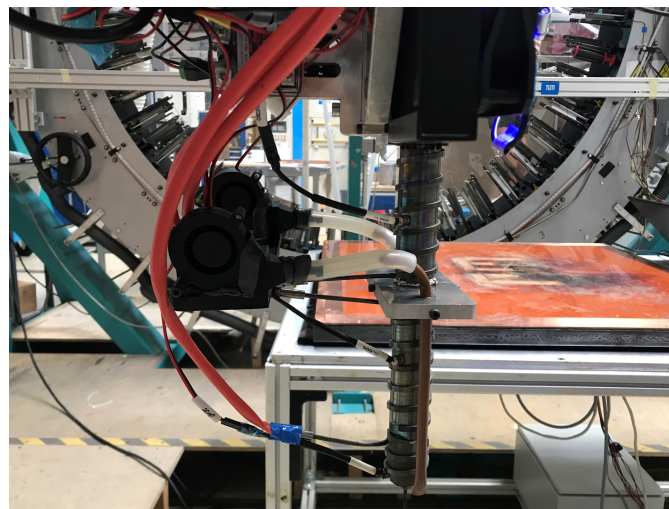


Figure 3.8 : Final orientation of the top part

The pulsar extruder has a tri-zone heating control, which can be seen in Figure 3.7. This ensures that the filament that goes through is at constant temperature at all times. The upper section receives the cold pellets and generates more heat to melt them, the middle section stabilizes the polymer at a precise temperature and finally the heater at the nozzle ensures an even flow. The heating up of each zone which is achieved by spiral heaters that are wrapped around the entire cylinder. These spirals have to be well attached and in full contact with the cylinder otherwise the cylinder won't be heated. To ensure that the filament is being melted temperature sensors are placed at each section at 0.5 mm from the molten polymer.

Before starting to print we had to check that the temperature distributed itself evenly throughout the whole extruder, to do this we started to heat up the spirals and with the sensor we checked the temperature. We saw that some parts of the spirals turned red, which means that they were not heating up the cylinder but heating up itself. The main reason for this to happen was that the spirals were not at full contact with the cylinder. In order to attach well the spirals to the cylinder and to maintain the heat in the cylinder and not dissipate it we used glass fiber to cover the whole cylinder and fixed it with a metal fixing band. This can be seen in Figure 3.9 which shows how the final structure of the cooling system looks like.

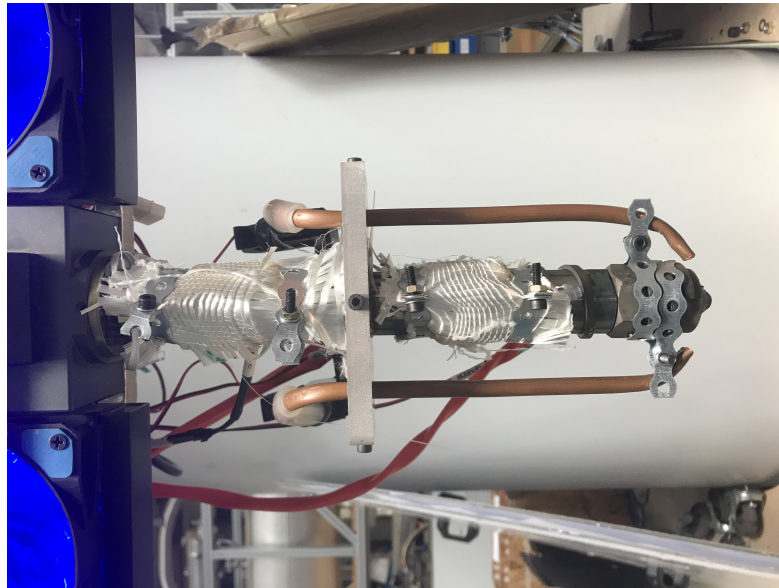


Figure 3.9 : Final position of the structure

With this final configuration we started to heat up the extruder again to check that all the parts were heating up correctly and evenly. Finally, everything was working properly and we were ready to print.

4 Material selection

When 3D printing, it is very important to take into consideration the material that will be used to print. The characteristics of the material chosen can influence the behaviour of our printed structure. A white paper written by Stratasys Direct Manufacturing called "3D printing, choosing the right material for your application" [10], shows a guideline to follow in order to choose the right material. This paper divides the selection process into four big categories: Application, function, geometry and post processing.

In our case, since we are working on a proof of concept, the application, function and postprocessing are not very relevant yet for the material selection. However, if the process works and we want to apply it to a specific engineering field, we would have to take these categories into account, and make sure that the material selected is compatible with them. Nevertheless, the geometry of the part is very relevant. We are trying to print lattice structures, which means that we will be printing in mid air, therefore, we need to choose a material that is able to cool off quickly and to attach easily to other layers being printed above.

4.1 Thermoplastics

In FDM the most used types of filaments are thermoplastics, they become malleable, and flexible when heating them up and they solidify when cooled down. This type of material provides rigidity and strength and high temperature tolerance which is what we are looking for.

- Semi-crystalline thermoplastics: In this category are included thermoplastics such as: Polyetheretherketone(PEEK), Polyethylene Terephthalate (PET), Polypropylene (PP), nylon (PA)... They have great bearing capabilities and high resistance, however, they are difficult to thermoform and bond. They have a sharp melting point, which means that in order for the plastic to melt and become malleable, it has to get to a specific temperature. For our application we need a thermoplastic that can easily bond with already printed parts, for example the vertexes of the lattice cells need to be easily bonded with the next cell. Also, they have high shrinkage and shrinkage anisotropy, which means that the printed parts will tend to crack and crumble and not stick to the table easily.
- Amorphous thermoplastics: materials such as Acrylonitrile Butadiene Styrene (ABS), Polycarbonate(PC), Polyethylene Terephthalate Glycol (PETG) are included in this category. Amorphous materials become soft and malleable over a broad range of temperatures, due to their high glass transition temperature. This is the

temperature of amorphous polymers at which increased molecular mobility results in significant changes in the thermal properties [4]. They thermoform easily and bond well [1]. This type of material could be a good possibility for our application since we are looking for a material that sticks into other layers easily and it can be quickly solidified.

For printing large lattice structures we will make transitions between solid and viscous state. Therefore, materials with quick transition time will work well. Next, I will analyse some materials that could fit the desired characteristics.

4.1.1 ABS

ABS is an amorphous thermoplastic polymer that has great material properties such as lightweight and good impact strength. The glass transition temperature is around 105°C. An usual problem that arises when using ABS is the dimensional stability and the large thermal expansion coefficient. Using fibers can mitigate this issue. Carbon fiber filaments are tiny fibers that are infused into a base material to improve the characteristics of that material. They are extremely strong and cause an increase in strength and stiffness. The fibers will add the dimensional stability that we need and prevent the shrinking of the part as it cools.

Using ABS with carbon fiber reinforcement would be a good possibility. It is able to solidify quickly when coming out of the nozzle and since it is an amorphous polymer it will bond easily with another printed structures.

4.1.2 Polycarbonate

PC is also an amorphous thermoplastic polymer. It has great heat resistance, strength and stiffness. Its glass transition temperature is around 147° which is very high. This means that it needs high temperatures to soften and flow. Since it comes out of the nozzle at such a high temperature, when coming out and being in contact with room temperature it will cool down very quickly which is just what we need. Nevertheless, this means that we need equipment that is able to withstand this temperature.

Similarly to ABS, PC is also prone to warping and shrinking but this can be mitigated with fiber reinforcement.

4.1.3 Polypropylene

PP is a semi-crystalline thermoplastic that comes from propene. It is relatively cheap and has very good mechanical properties. However, due to its semi-crystalline nature,

it will tend to shrink and warp during printing, and the structure will lose accuracy. As the material cools down, the volume of the polymer decreases, as long as the material is above the glass transition temperature, the material will shrink. The shrinkage will be far more than the one experimented with amorphous polymers such as ABS.

The part will shrink in an anisotropic manner at different position, due to insufficient or nonuniform cooling, which will make it pron to warpage[7].Several studies have been made to improve the dimensional inaccuracy.The main idea is add fillers into the thermoplastic [15] which will make it a good candidate for 3D printing. All these characteristics make it a good candidate for 3D printing.

4.2 Final Selection

Initially, we wanted to use ABS with carbon fiber reinforcement but due to the actual situation, the delivery would take several weeks. We had the same problem with PC with carbon fiber reinforcement. Given the situation, we had to chose a material that was available at the work shop, which was PP with 30% glass fiber reinforcement. We know that it is not the ideal material but for the proof of concept it should be okay.In the future it would be interesting to see how the result will change if we use one of the other materials.

5 Lattice structures

Lattice structures are the future of engineering. They are three dimensional unit cells made of nodes and struts, that joined together reduce the weight of the designed piece significantly without sacrificing strength or structural integrity. Strong and lightweight components are needed in many engineering fields to increase the structural efficiency of the parts[9].

5.1 Advantages of lattice structures

- Structural advantages: Lattice structures use with low material the most amount of space. This means that they cover a lot of surface without cost expense. Depending on the shape of the cell we will get different mechanical benefits[13]. But overall, lattice structures offer great possibilities, such as a greater shock absorption, noise damping or even the impact stress can be reduced.
- Strength-to-weight ratio: Using lattice structures means that the total material used is reduced. Reducing material also implies a reduction on the overall strength of the part, but an improvement on the strength-to weight ratio [16].
- High surface area: Even though that the material is reduced, lattice structures enclosure large amounts of surface area, which facilitates heat exchangers.
- Shock absorption: one of the big advantages of using lattice structures is that they are excellent absorbing impacts. They can be inserted into fragile components so in case of dropping them or failure, they can absorb the shock energy [12]. Also, since they are great at shock absorption, they can be used as dampers. They can reduce vibrations and diminish the amount of energy that is going into the system.

More benefits can be obtained if the correct design for additive manufacturing considerations are taken. Considerations such as cell structure, material selection or orientation of the fibers are important and could have a great impact in the final properties of the printed structure.

5.2 Applications

Lattice structures offer a wide range of possibilities in different engineering fields such as aerospace and architecture.

5.2.1 Aerospace

Due to the light weight reduction that lattice structures present, they have great potential in the Aerospace sector, where lightweight is crucial. The replacement of solid internal parts with lattice structures would suppose a reduction in weight and an addition of some internal characteristics such as increase of strength.

There is research being done towards this field, such as Zhoe et al. The thermal properties and advantages that lattice design offers have been put into practice to design a lightweight phase-change thermal controller based on lattice cells [17]. This type of controller can be used to oversee the temperature distribution of the electronics of in the aircraft.

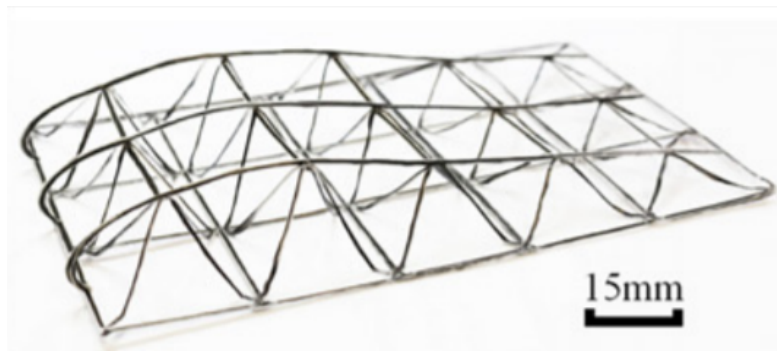


Figure 5.1 : Integral wing [8]

Wings made up of lattice structures are already being developed like the one seen in Figure 5.1, which was done with the free-hanging 3D method for continuous fiber reinforced thermoplastic [8], such a complex structure verifies the flexibility for this printing method, the overall structural strength was considerably improved but there were some voids in the cross sectional micrographs of the fibers in the lattice cell units. Nevertheless, this is a great leap for free form printing in the aerospace field.

5.2.2 Architecture

As it has been said before, lattice structures offer great advantages in the construction world. One of the main inspirations for this project was the company Branch Technology [2], which is an architectural company that is focused on the large scale 3D printing. They use a process called Cellular Fabrication, a 3D printed matrix made of lattices of ABS reinforced with carbon fiber. These lattices can later be used as modular wall systems and integrated into common building materials as seen in Figure 5.2. This allows to construct large structures with less material and to increase the overall structural strength.

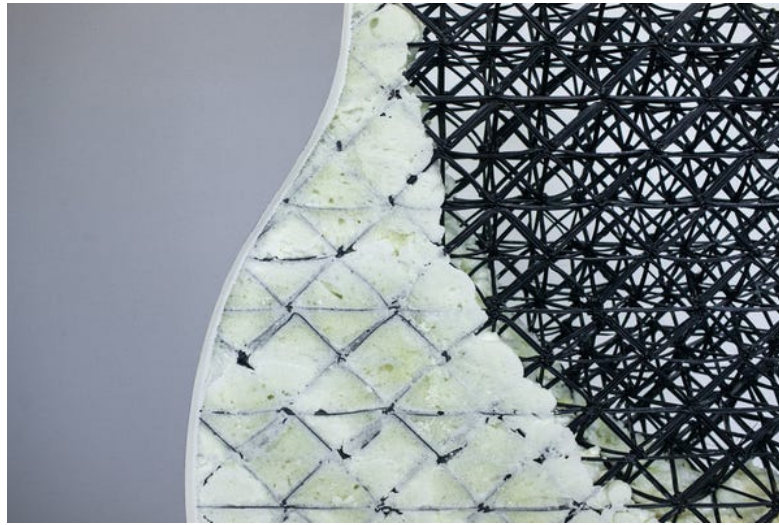


Figure 5.2 : Cellular Fabrication. Image courtesy of Branch Technology

5.3 Generation of the lattice structures

To design the overall structure of the lattice cells, Grasshopper was used. Grasshopper is visual programming language that is installed as a plug in for Rhinoceros 3D, a CAD program. Thanks to Grasshopper we are able to reference geometries created in Rhino and vice-versa. The main advantage that Grasshopper presents is that it allows you to do a completely parametric design, this means that once you have created a lattice structure, you can change its shape and size without any problem.

In order to prove the concept of free form printing lattice structures, 3 structures were created: one simple cube, a larger horizontal panel divided into small lattice cubes and finally a "cylinder" structure. The first two examples use only three axis of the six axis robot, they only use the x,y,z directions. Whereas the last example uses 5 axis, in addition to the three previously mentioned axis we will try to use the a,b coordinates, the rotation of the x and y axis.

Once the G-Code was available, the program RobotDK was used to simulate the movements of the KUKA robot to check that everything was working correctly before printing. In order to print with the KUKA robot arm, the interface used to send the commands to the robotic arm was RobotDK.

Each design was done independently in Grasshopper and its workflow will be explained in detail in the following subsections.

5.3.1 Simple cube design

This design was the simplest of the three. With this structure what we want to prove is that free form printing a simple lattice structure such cube is feasible.

Using Grasshopper to write the code for the piece was very useful since while I was designing the structure I could check that everything was working as planned in the Rhino interface. Inside Grasshopper, another plug in was used to write the G-Code; Silkworm. Silkworm translates Grasshopper and Rhino geometry into G-Code for 3D printing. It let's us manipulate the settings of the printer to get the desired characteristics of the printed geometry.

The workflow followed for the design of the simple cube can be seen in Figure 5.3. Firstly a simple box was created with the command *Box*, the dimensions are fully parametric, and can be changed as pleased. We decided to do a cube of 150x150x150 mm, big enough to ensure that there were not going to be any collisions when printing. This is why in the Box the x,y,z have a slider of 75 mm, since it is going to extrude 75 mm in the positive and negative x direction and so on...

Once the box has been created, the different elements of the cube can be obtained when using the command *Deconstruct Brep*. This gives us the faces, edges and vertices of the cube. We are interested in obtaining the vertices of the cube, because we will use them to set the order of printing. 5.4.

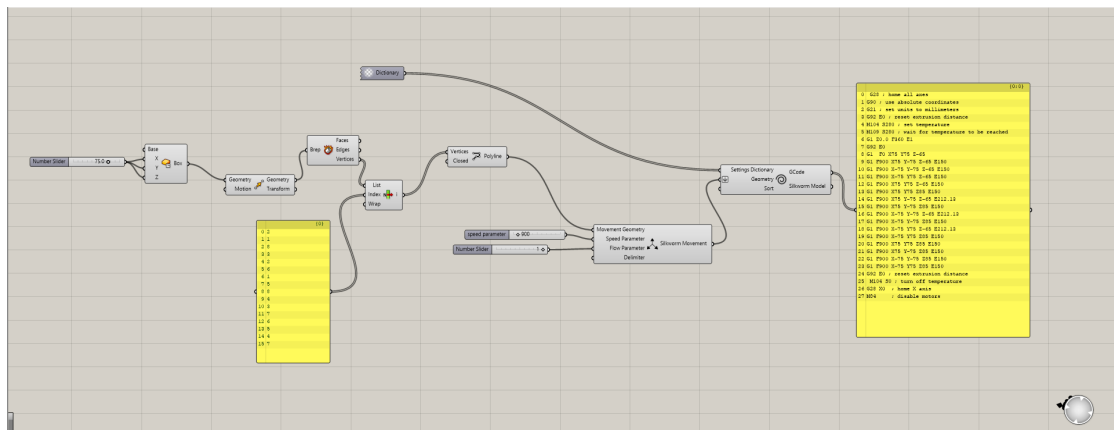


Figure 5.3 : Grasshopper workflow of simple cube

We want to print the whole cube with a single polyline, without having to stop extruding the filament. Therefore, we have to tell the extruder the order of printing. Thanks to the command *Deconstruct Brep* we have assigned to each vertex an integer number. what we have to do next is establish the correct order of the vertices to be able to make a polyline. This is done with the command *List item*, that filters the list of vertices of the cube and rearranges the vertices in the correct order set with panel of indexes. Once this is done, with the command *polyline* we join all the vertices together to get the desired

polyline. The result of such polyline can be seen in Figure 5.4 which shows the Rhino interface.

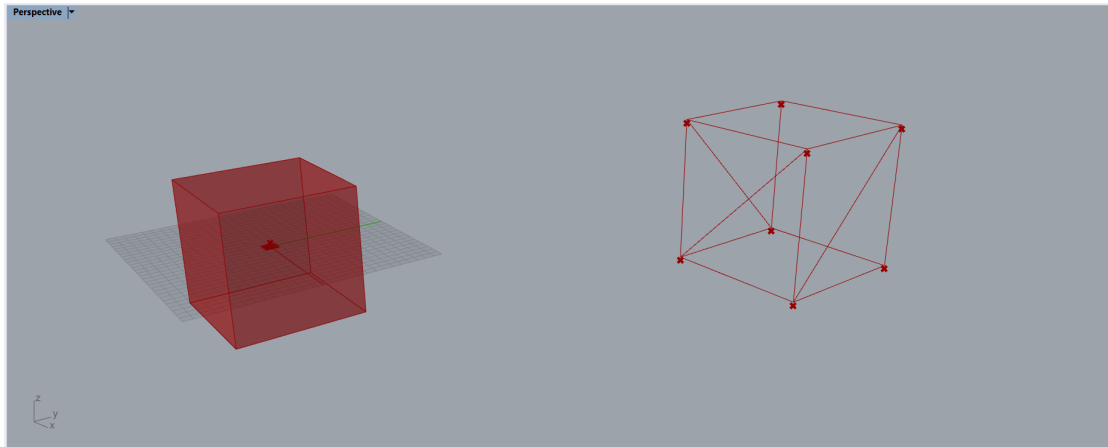


Figure 5.4 : Simple cube in the Rhino interface

Once we have the polyline we can use the plug in Silkworm to transform the desired geometry into G-Code. We use the command, *Silkworm Movement* to input the movement geometry, this will be the geometry that will be described by the toolpath, which is the polyline. We can also input the speed parameter and the flow parameter which have been set to $900\text{mm}/\text{min}$ and 1mm^2 respectively. The output of this command is the movement of the toolpath. Finally this movement is inputted into the *Silkworm generator*, which outputs the G-Code. In order to get the G-Code we have to input the Settings of the extruder we will be using. This information is stored in the dictionary, the information that was included in this settings dictionary can be seen in Figure 5.5. Finally, the G-Code can be seen in the output panel, which is seen in Figure 5.3, this is automatically saved and will later be used in the printer. .

```

absolute_extrudersteps = 0
acceleration = 0
bed_size = 800,800
bed_temperature = 100
bridge_fan_speed = 100
bridge_flow_ratio = 1
bridge_speed = 20
brin_width = 3
complete_objects = 0
cooling = 1
disable_fan_first_layers = 1
duplicate = 1
duplicate_distance = 6
duplicate_grid = 1,1
end_gcode = M104 S0 ; turn off temperature/M28 X0 ; home X axis/M984 ; disable motors
external_perimeter_speed = 100%
extra_perimeters = 1
extruder_clearance_height = 20
extruder_clearance_radius = 20
extruder_offset = 0x0
extrusion_axis = E
extrusion_multiplier = 1
extrusion_width = 4
fan_always_on = 0
fan_below_layer_time = 60
filament_diameter = 2.95
fill_angle = 45
fill_density = 0.4
fill_pattern = rectilinear
first_layer_bed_temperature = 100
first_layer_extrusion_width = 100%
first_layer_height = 100%
first_layer_speed = 20%
first_layer_temperature = 280
infill_acceleration = 50
infill_every_layers = 1
infill_extruder = 1
infill_extrusion_width = 3
infill_speed = 20
layer_gcode =
layer_height = 3
max_fan_speed = 100

infill_extruder = 1
infill_extrusion_width = 3
infill_speed = 20
layer_gcode =
layer_height = 3
max_fan_speed = 100
min_fan_speed = 35
min_print_speed = 20
nozzle_diameter = 3
output_filename_format = [input_filename_base].gcode
perimeter_acceleration = 25
perimeter_extruder = 1
perimeter_extrusion_width = 3
perimeter_speed = 20
perimeters = 3
post_process =
print_center = 400,400
randomize_start = 1
retract_before_travel = 0
retract_length = 0
retract_lift = 0
retract_restart_extra = 0
retract_speed = 20
rotate = 0
scale = 1
skirt_distance = 6
skirt_height = 1
skirts = 0
slowdown_below_layer_time = 15
small_perimeter_speed = 20
solid_fill_pattern = rectilinear
solid_infill_speed = 20
solid_layers = 3
start_gcode = G28 ; home all axes
temperature = 280
top_solid_infill_speed = 20
travel_speed = 20
use_relative_e_distances = 1
xbar_heightfromnozzletip = 30
xbar_width = 30
z_offset = 0

```

Figure 5.5 : Settings dictionary of the extruder

As it has been said before, the code generated will extrude a cube. Firstly, the base of the cube will be printed and then the extruder will go upwards and downwards doing a

diagonal, and then upwards again and so on, until coming up to the last upwards edge where instead of going down into the diagonal it will go straight to the next vertex, making the top square of the cube. This can be seen in Figure 5.4.

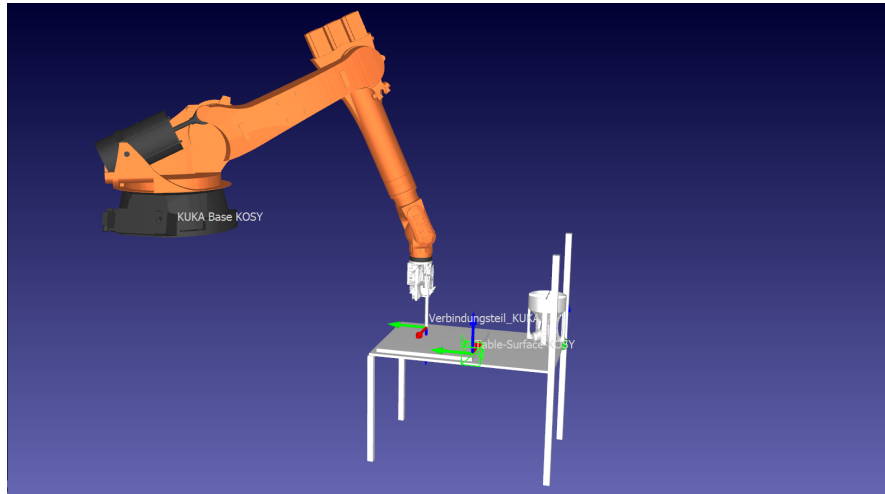


Figure 5.6 : Grasshopper workflow of simple cube

To check that everything worked as predicted, we load the G-Code generated in Grasshopper into a RobotDK program that has the KUKA robot and the table where everything will be printed on. We loaded the first code of the cube and the result can be seen in Figure 5.6. The cube that will be printed is displayed as a collection of green lines. As it can be seen, the cube is not on the table, it is a little bit outside the table and it even intersects the table. We cannot send this code to the extruder because the robotic arm would collide into the table and break it. Therefore, we have to go back to Grasshopper and move the geometry.

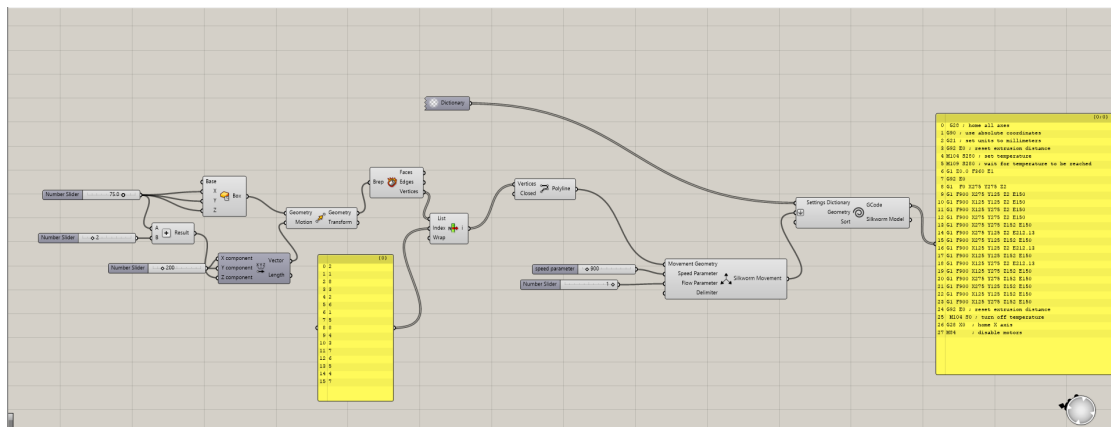


Figure 5.7 : Grasshopper workflow of simple cube

In Figure 5.7 it can be seen how we have added some new commands. It is important to mention that the bed size of the table is 800mmx800mm. Firstly we have created a vector that moves upwards half length of the edge and 2 mm more, so the extruder does

not collide with the table but stops 2 mm before. In the x and y direction we have added 200 mm so it is somewhere inside the table. This vector has been used with the *transform* command and the output geometry will now be deconstructed and so on which is done and saved automatically.

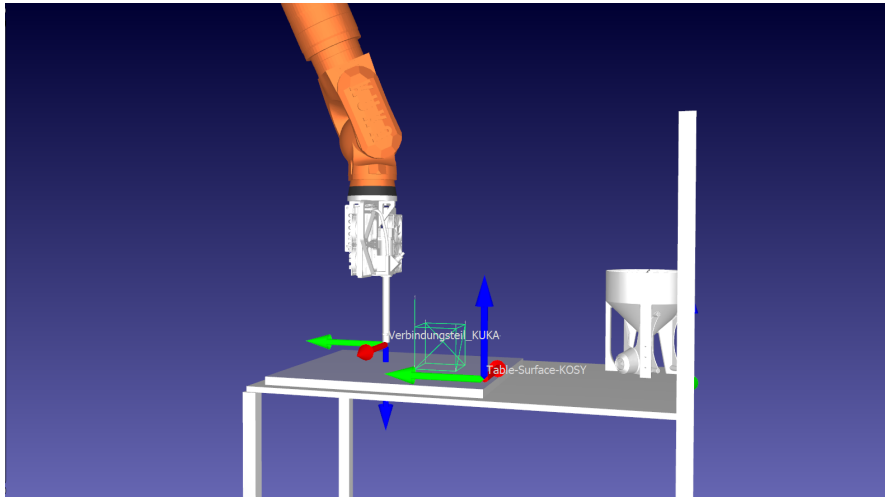


Figure 5.8 : Grasshopper workflow of simple cube

With this new G-Code we go again into RobotDK and load the file. As it can be seen in Figure 5.8 now the cube is inside the table parameter and it is 2 mm above it, therefore, we are ready to print it.

5.3.2 Horizontal panel design

To prove the free form printing concept, a bigger piece was designed, a horizontal panel. In Figure 5.9 it can be seen the whole workflow that was created. It is a bit more complicated than the single cube but the logic is the same. We are still using grasshopper as a plug in with Rhino and Silkworm to write the G-Code for the print.

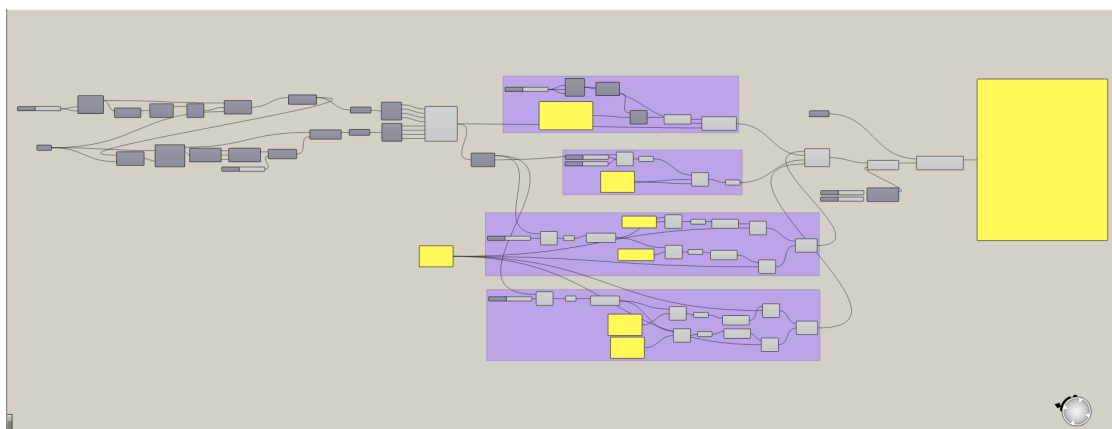


Figure 5.9 : General workflow of the horizontal panel

First of all, we have to create a surface in the Rhino interface to be able to manipulate it and create the lattice structure. In Figure 5.10 it can be seen such surface that it has the dimensions of 430 mm x 330mm, since it would be a good size for the printing bed. It is important to remember that the code is fully parametric.

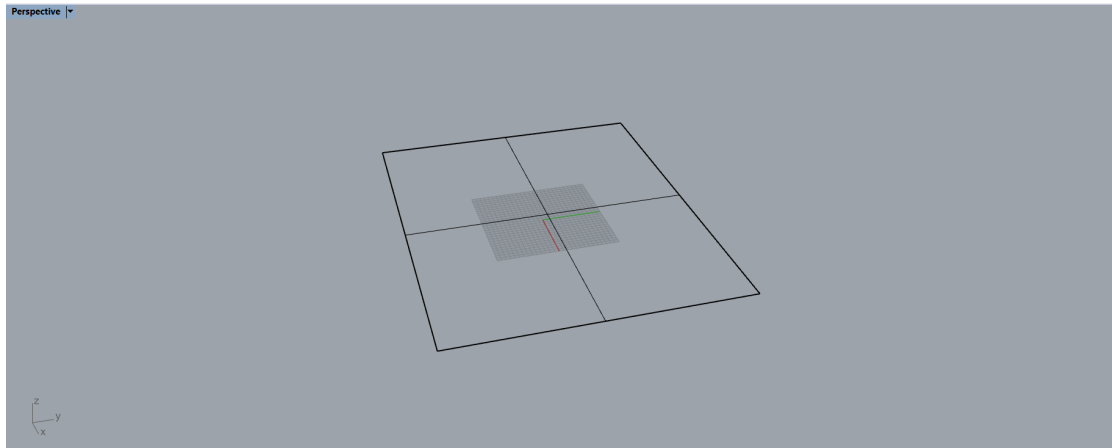


Figure 5.10 : Horizontal panel in Rhino.

Once the surface has been created we can reference it in Grasshopper, using the command *Surface* which can be seen in Figure 5.11 and we can start modifying this surface to create the lattice structure we are looking for. Even though that we have a 2D surface and that we will have to transform it into a 3D object, we can start dividing it into the small squares that will make up the lattice structure. This procedure is seen in Figure 5.11. We firstly create a square, with a extent of 4 in the x and y direction, this means that we will obtain 4 square cells in the x direction and 4 in the y direction. This can be changed to any number. Since this square is of random dimensions and we will want to morph it into the already created surface, we use the *bounding box command* to create a reference of this geometry. Next up we deconstruct this box to get the corresponding faces, edges and vertices with the command *DeconstructBrep*.

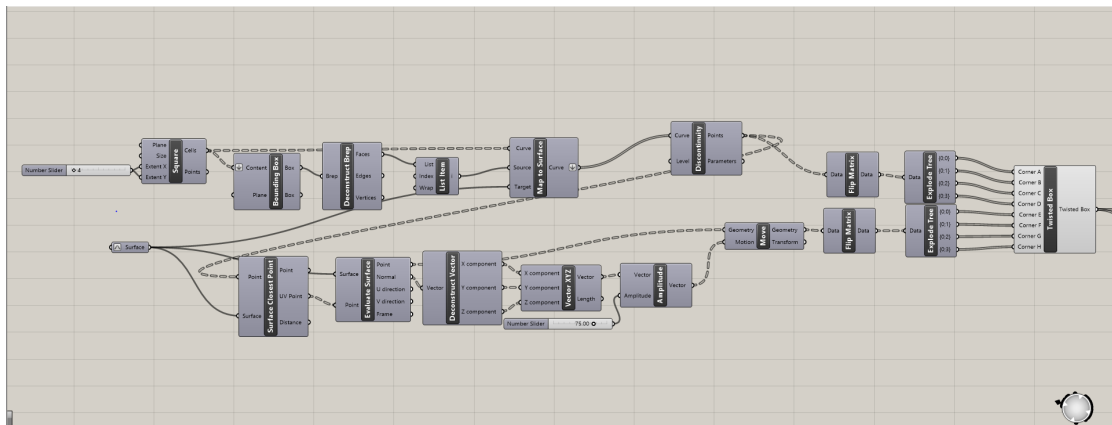


Figure 5.11 : Division of the horizontal panel.

Finally we only have to use the command *Map to surface* to map the faces of the deconstructed Brep into our Rhino surface, the result can be seen in Figure 5.12, the initial surface has been divided into 4x4 squares, rectangles in this case since the surface is not a square.

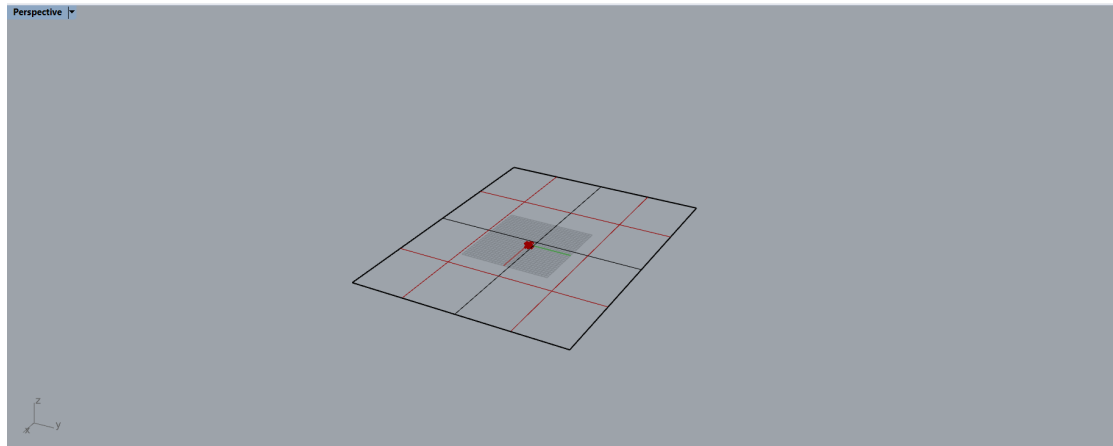


Figure 5.12 : Deconstructed horizontal panel into small squares

Nevertheless, the *Map to Surface command* gives us as an output a curve, and we want to have the collection of points, to be able to join them in small segments and then print them. This is done with the *discontinuity* command which gives us the collection of points found along the curve. Once we have the horizontal panel divided into small rectangles we can extrude them in the vertical direction to obtain a 3D structure. To do this we take the points of the discontinuity and use the *surface closest point* command, with the points and the rhino surface as inputs. This command gives us the *uv* coordinates of the points, with this we can evaluate the surface and get the normal component to construct a vector in the normal direction. To get the desired height of the panel we use the command *amplitude* which has a slider of 75mm of height but can be changed parametrically.

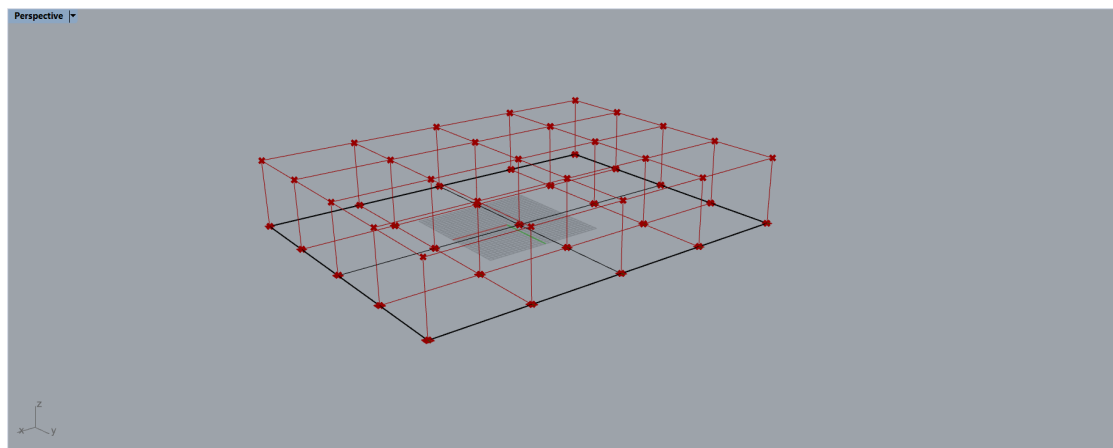


Figure 5.13 : Twisted box

We have the lattice structure created, now we have to create the toolpath for the extruder to print it. Since it is a more complex structure, it makes sense to divide it into different subparts to make it as efficient as possible. Firstly the bottom part of the structure will be printed, then the middle part, made up of different cubes and finally the top section.

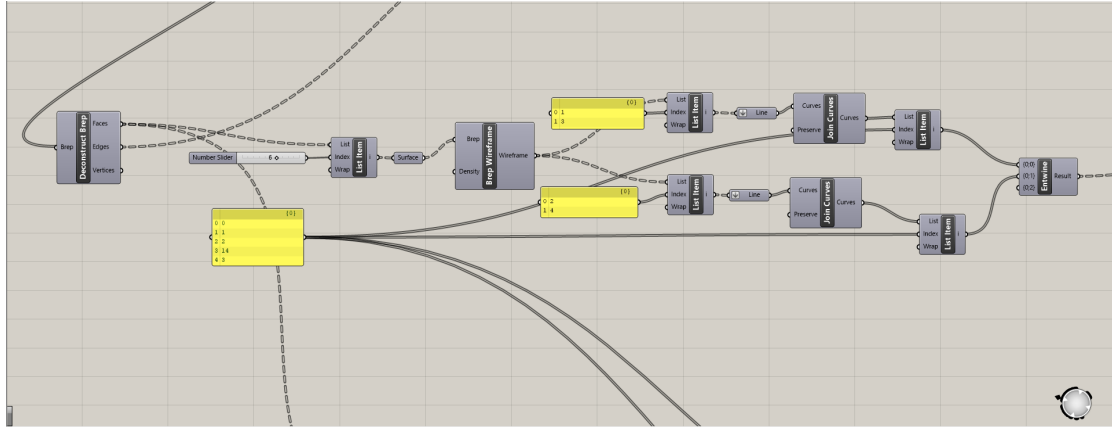


Figure 5.14 : Grasshopper code for the bottom part of the panel

For the bottom and top part the same logic was implemented since they are the same pattern but only with an offset, this is seen in Figure 5.14. Firstly, we deconstruct the twisted box using *deconstruct brep*, we are interested on the faces, on the bottom face, this is why we use the list item and a slider to choose the correct face. Then, we *transform* the face into a surface to be able to manipulate. We use the command *Wireframe* to extract the wireframe from a Brep. The result can be seen in green in Figure 5.15.

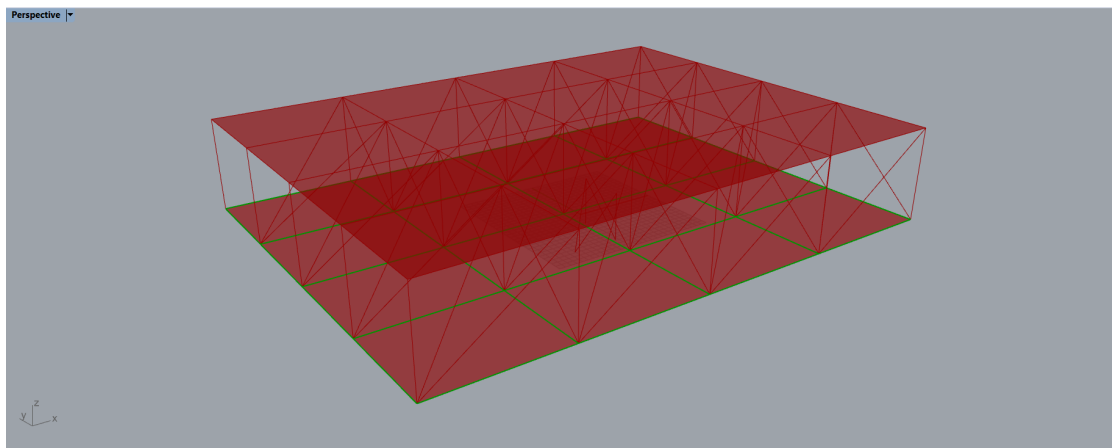


Figure 5.15 : Bottom part toolpath seen in green

One could think that this wireframe could be the input target of the Silkworm generator, but it turns out that wireframe command runs 64 times, which means that we have 64 lines, therefore, if we were to use this as an input for the Silkworm generator, the extruder would go more than twice through the same point. In order to avoid this I will filter the wireframe again using the *list item* command again, this will be done twice, once in the vertical direction and then in horizontal direction.

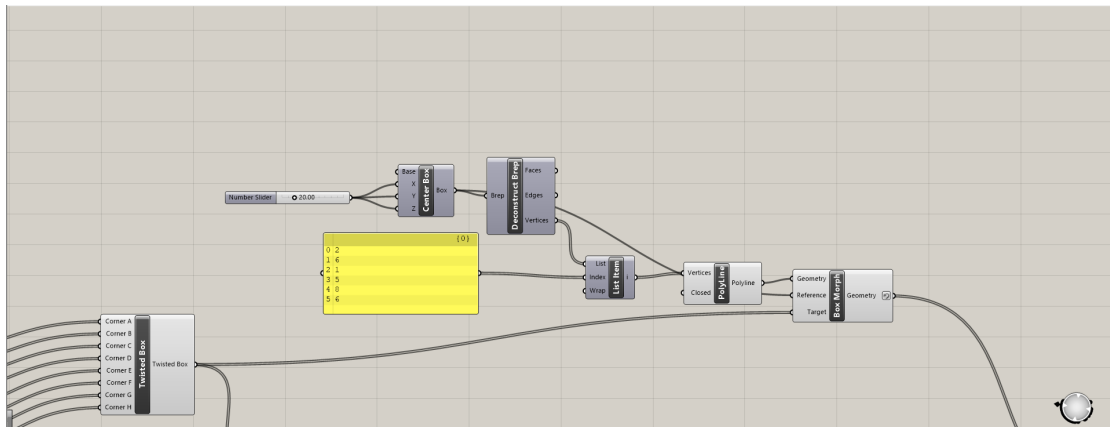


Figure 5.16 : Simple cube in the Rhino interface

Once again we keep getting the same problem, which is that the output from the command line gives us 32 different lines. We only want 5 of those 32 lines, otherwise we would keep printing through the same point more than once. To solve this we filter once again the command *line* with *list item* and we finally have 5 lines as output. To join the vertical and horizontal segments in a single command we have used *Entwine* which flattens and joins two strings of data without altering the order. The same workflow is done for the vertical part of the panel.

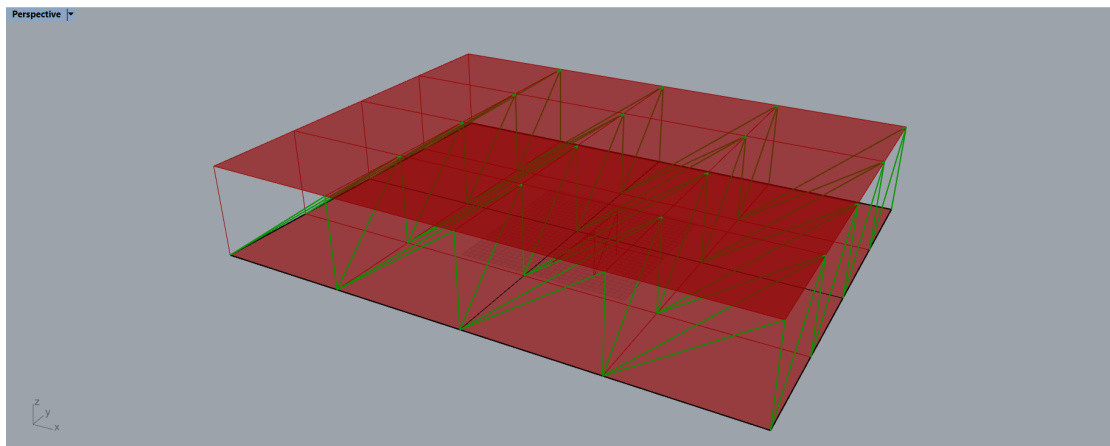


Figure 5.17 : Middle section of the horizontal panel

For the middle part we have used the same toolpath that for the simple cube, as it can be seen in Figure 5.16 we create a center box or a random dimensions and *deconstruct the brep* to obtain the vertices of the box. Then we use *list item* to arrange the vertices, the order in which they will be printed. The output of the list item is transformed into a *polyline* and finally we use the *box morph* command to morph the boxes created into our twisted box. The result can be seen in Figure 5.17, the green part is the toolpath.

If we look closely at the image it can be seen that at the left part of the panel the vertical struts are missing. This is because when choosing the order of the vertices we

did it in such a way that the extruder would not go twice through the same point. We need to modify the code to ensure that the struts are printed.

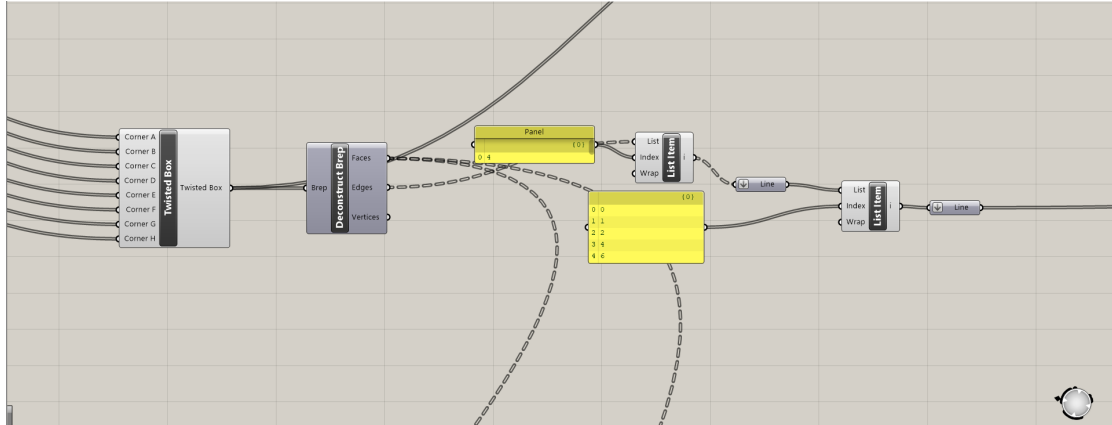


Figure 5.18 : vertical struts in the grasshopper interface

In Figure 5.18 can be seen the code implemented for the remaining vertical struts. From the *deconstructed Brep* of the twisted box we obtain the edges of the horizontal panel we filter them using the *list item* to get the vertical struts that we turn them into *lines*. Again, we encounter the same problem as before, since the output of the command lines gives us 32 lines, which means that the extruder will go more than once through the same point, colliding with already printed structure. To solve this problem we filter the lines with *list item* and we finally get 5 lines as an output which is what we wanted.

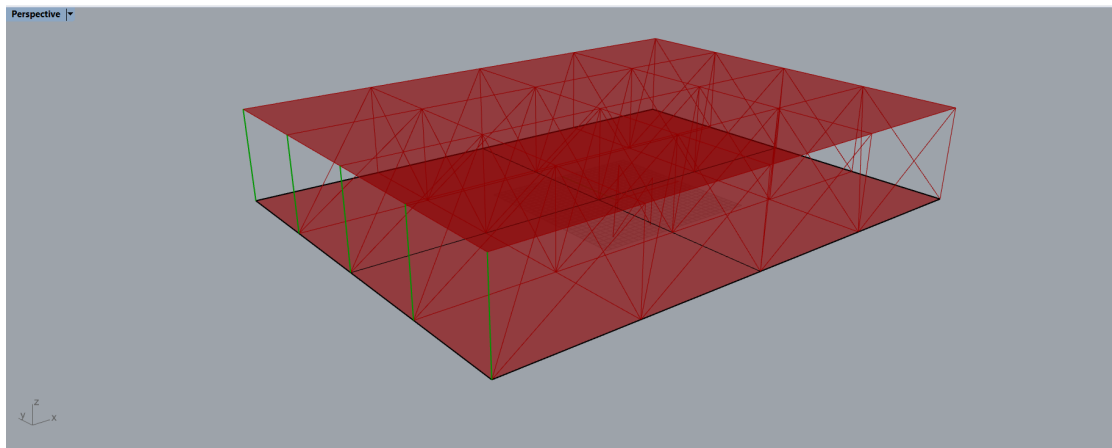


Figure 5.19 : Vertical struts of the middle section

In Figure 5.19 the vertical struts that I have added to the code are seen. Now the whole middle section will be printed and the top part will be able to stick correctly. For the top part of the panel the same logic of the bottom panel was implemented but with the corresponding vertical offset.

Now that we have all the subsections divided and its corresponding toolpaths, we can create the G-Code. In Figure 5.20 the logic to generate the G-Code can be seen. It is

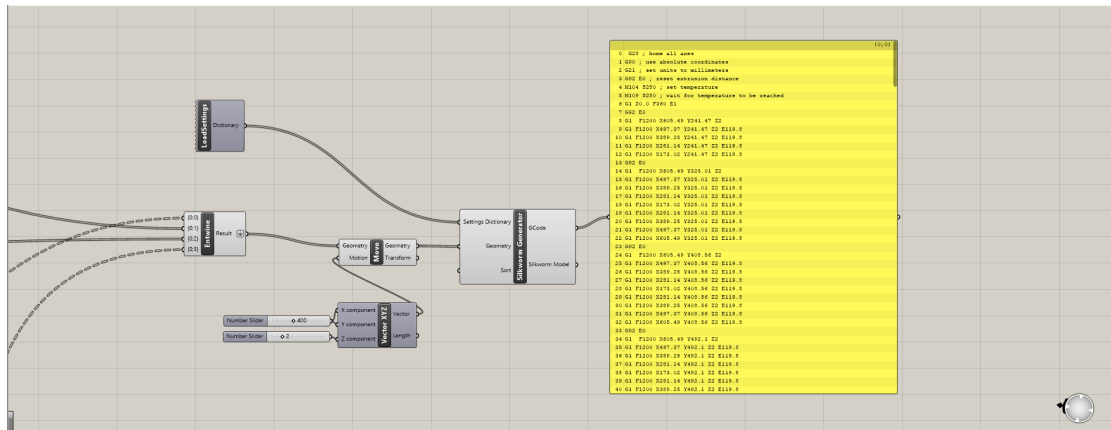


Figure 5.20 : G-Code generation of the horizontal panel

similar to the simple cube structure, but a little bit more complex. Firstly we have to take all the toolpaths in the correct order and merge them into a single command. This is done with the command *Entwine*, next up we move the structure into the desired space so it is placed on the table. It can be seen in Figure 5.20 that the offset for the x and y axis is of 400 mm so it's inside the table and for the z axis is 2 mm so the extruder does not collide into the table. This is completely parametric.

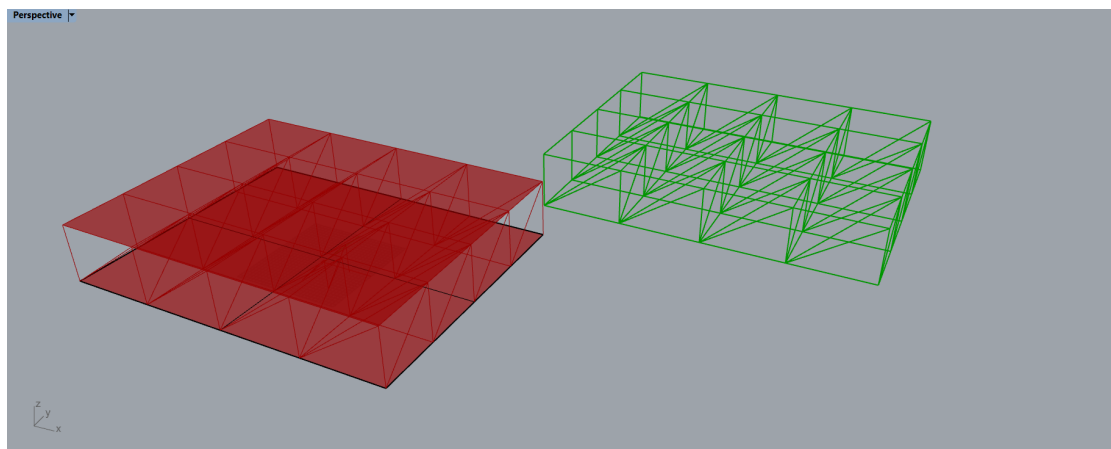


Figure 5.21 : Horizontal panel final toolpath

Finally what is left to do is to use the *Silkworm generator* and generate the G-Code with the same settings dictionary that the simple cube seen in Figure 5.5. The G-Code generated can be seen in the yellow panel of the Figure 5.20, it can be automatically saved and used in the desired printer. The final toolpath of the horizontal panel in the Rhino interface can be seen in Figure 5.21, which is the structure ready to be printed. The green structure is the already moved part so its inside the printing table.

In order to validate that the G-Code generated worked properly on the extruder we load the G-Code generated in Grasshopper into the RobotDK program that has the KUKA robot and the table where we will be working on. In Figure 5.22, it can be seen how the

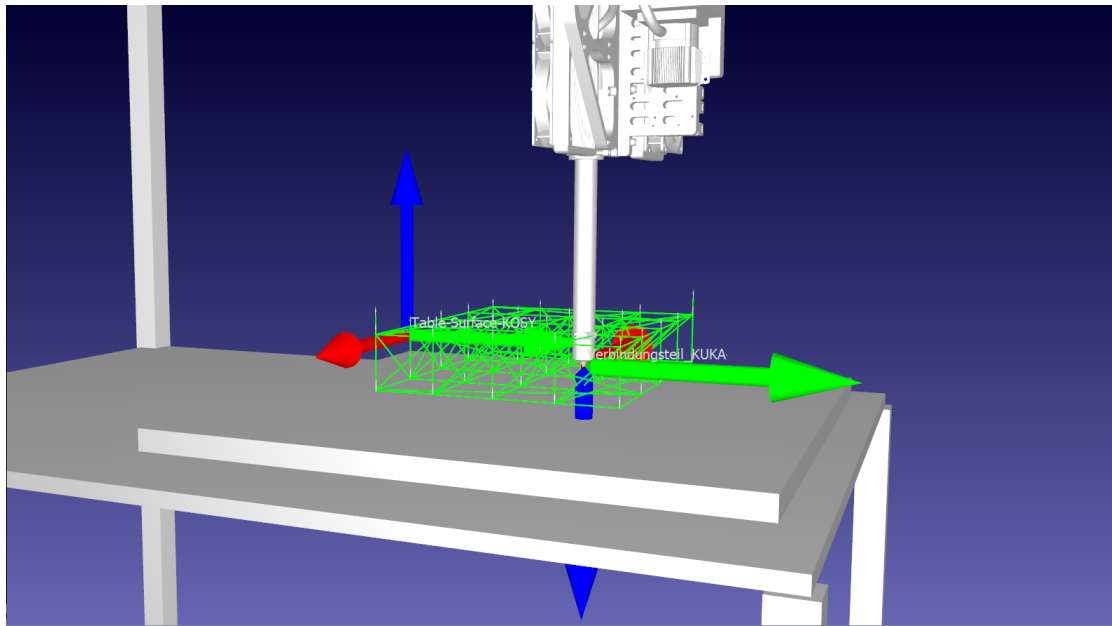


Figure 5.22 : Horizontal panel in RobotDK

part is on the table as we expected, more or less centered on the table. We also check that there will not be any collisions running the preprocessor and that everything will be correctly printed. Once this is done we can try to print the part and verify the code G-Code created.

5.3.3 Plane orientation design

The previous two structures designed prove the free form printing concept without twisting and rotating the robotic arm. In theory they could be printed in a normal printer, since no rotations are required. In order to go a little deeper, it would be great if we could print a structure that required the tilting of the robotic arm, in other words the rotation of the x and y axis, which in G-Code it translates to a and b values. We want to avoid the rotation of the z axis, c value in G-Code since the cables of the extruder could be broken or unplugged when rotating this axis.

The Silkworm plug in that we have used for the previous examples cannot be used anymore since it does not allow rotation values. The output of this plug in cannot give a,b,c values. Therefore, we have to find another way to generate the G-Code. From the previous examples it can be seen that the design of a G-Code is very simple, what is important is to have the right coordinates, once we have the coordinates we only have to add a couple of commands to generate the G-Code.

Grasshopper is rapidly developing and every day new and exciting plug ins are created to fulfill the needs of the community. Looking for a way to create a toolpath which could make the KUKA arm rotate I came across a plug in called KUKA|prc (parametric robot

control)[3]. This plug in uses the parametric environment of grasshopper to immediately generate toolpaths in KUKA robot language (KRL) exported as *.src files. The commands given to the robot are given in a cartesian coordinate programming system seen in Equation(5.1), which means that the position of the KUKA robot is at $X=20$, $Y=20$, $Z=20$, the rotation of the x axis (A) is of 45° , the rotation of the y axis (B) is of 0° and the rotation of the z axis (C) is of 45° .

$$X20 Y20 Z20 A45 B0 C45 \quad (5.1)$$

In order to prove the free form printing with rotations concept we have to design a structure in which the toolpath changes the orientation of the robotic arm by using the KUKA|prc. We will establish the toolpath of the extruder as a succession of planes, the coordinates of the plane will be calculated and the extruder will move from one plane to the other. If the planes are positioned in such a way that the orientation changes, rotation values, A, B and C will appear in the *.src file.

With this new workflow in mind we can start thinking about what kind of structure we could print to prove this concept. It should be a simple structure, in which the movements of the robotic arm can be easily controlled. The figure I designed to prove this concept can be seen in Figure 5.23. It is composed of two circles, separated by an offset in the z direction, each circle is divided in 10 points, and a continuous zig zag line is created from one point of the bottom layer to a point of the top layer. In total the continuous line goes through 5 bottom points and 5 top points.

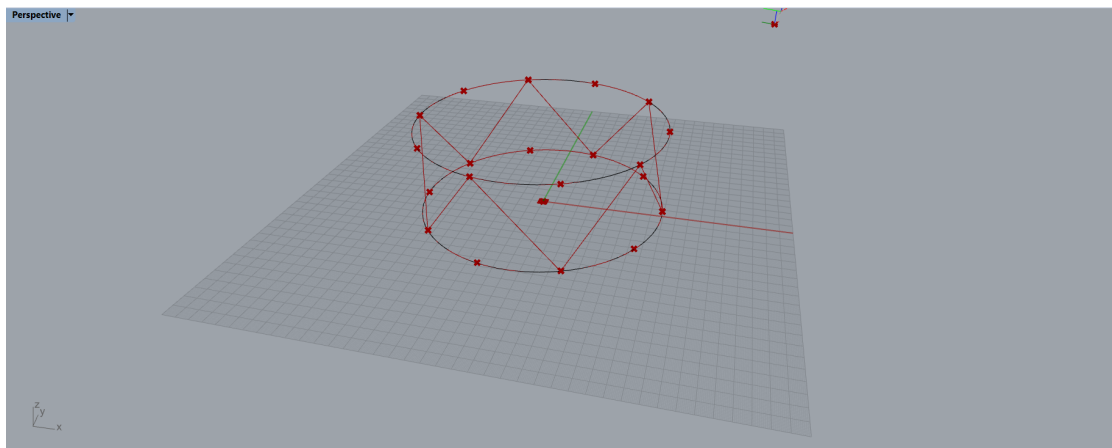


Figure 5.23 : Plane orientation geometry in Rhino interface

I intend to print the bottom five points as a continuous line, which will make the form of a pentagon. Then the continuous line of the up and down movement will be printed and finally the top layer, made up of a line that goes through the top points making another pentagon. Once the geometry has been defined we can start developing the code in the

grasshopper interface. The total workflow created can be seen in Figure 5.24. Now I will explain in more detail how the code was generated.

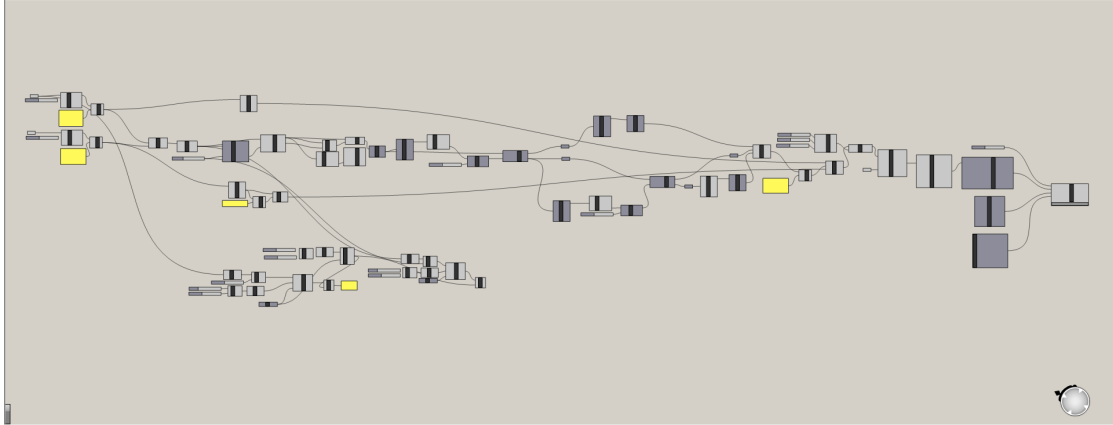


Figure 5.24 : Plane orientation workflow in Grasshopper interface

Firstly both circumferences are referenced in the Grasshopper interface, which is seen in Figure 5.25, using the command *curve*. Then each curve is divided into 10 segments using the command *divide curve*. The number of segments is parametric but for simplicity we have chosen 10. The command *divide curve* has different outputs, we are interested on the points, thanks to the command *list item* we are able to filter the points that we need for the up and down curve. Finally with the command *weave* we are able to merge the points from the bottom circumference to the top one and with the help of the command *polyline* we are able to join the points in a single line.

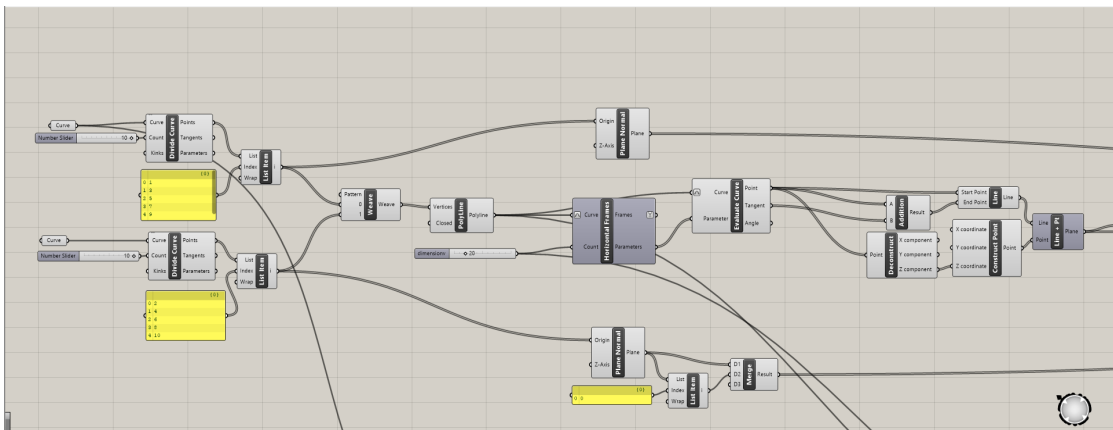


Figure 5.25 : Plane orientation workflow in Grasshopper interface

With the main design created we can start creating the planes that will make up the toolpath for the robotic arm. The bottom and top layer of the structure will be printed without any rotation of the axis, since it is a horizontal print, rotations are not needed, therefore, the planes created will be horizontal. However, for the up and down movement of the polyline, the robotic arm will change the orientation of the x and y axis, hence the planes created through these points will not be horizontal but tangent to the polyline.

We need to establish how many planes we want for the toolpath. We will need at least five planes that contain the five points of the bottom pentagon and another five planes for the top pentagon. For the polyline that connects the bottom and top layer we will need at least one plane in each segment. We use the command *horizontal plane* to divide the polyline into 20 horizontal frames, five at the top, five at the bottom and the remaining are equidistant in the connecting segments.

For the printing of the bottom and top pentagons the workflow is straight forward, we use the command *plane normal* with the points of the list item as origin to create the planes desired which are horizontal planes.

Once the bottom pentagon has been printed we proceed with the polyline. In this case we are looking for planes that are tangent to the line. To create such planes we use the command *evaluate curve*, with the polyline as an input and the parameter created with the horizontal frames as the other input needed for the evaluation of the surface. This command gives us the coordinates of the 20 points and the tangent vectors at those points. With this outputs we can construct the tangent planes that we were looking for. We create a line with the start coordinate of one of the points and the end point as the sum of that point and the tangent. We extract the z coordinates of all the points and we use them with the line to create the plane.

This gives us the tangent planes to all 20 points created, but we have to remember that for the bottom and top layer we don't want tangent planes, we don't want any tilting of the robotic arm therefore those planes have to be modified and turned into horizontal planes.

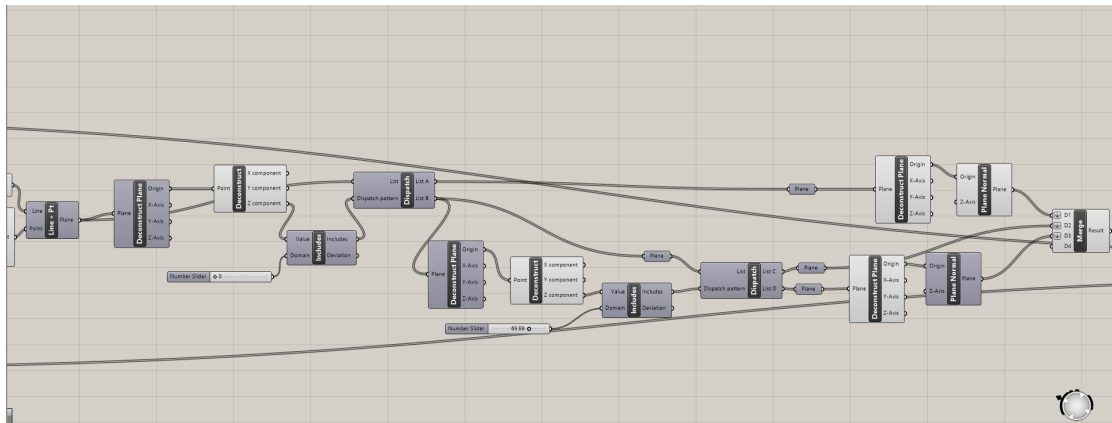


Figure 5.26 : Filtering of the necessary planes in Grasshopper

In order to delete the change the orientation of the bottom and top part that are tangent to the polyline I have implemented the code seen in Figure 5.26. We take the planes created and use the command *deconstruct plane* to get the origin of each plane and we deconstruct each origin into its corresponding coordinates using the command *deconstruct point*. Next up we use the command *includes* to filter the the points that contain

the value $z=0$, in other words the bottom points of the pentagon. Next up we use the command *dispatch* to divide into two lists the origin of the planes, in list A we get the points in which the z coordinate of the origin is 0 and in list B the remaining points, which is a total of 15 points. With list A we can already create the horizontal planes that we were looking for. List B will be filtered again to extract the points of the top pentagon, the same procedure is done but now the domain that has to be included is that the z coordinate has to be 69.88 which is the height of the pentagon. Once again, we dispatch the list of planes and we get another two lists, list C which is composed of the ten planes tangent to the polyline and list D in which the five points of the top layer are included, with these points we can create the horizontal planes we were looking for. Finally, the only thing left to do is merge the five bottom planes, the 10 planes of the polyline and the five top planes into a single list using the command *merge*. The result of this procedure can be seen in Figure 5.27

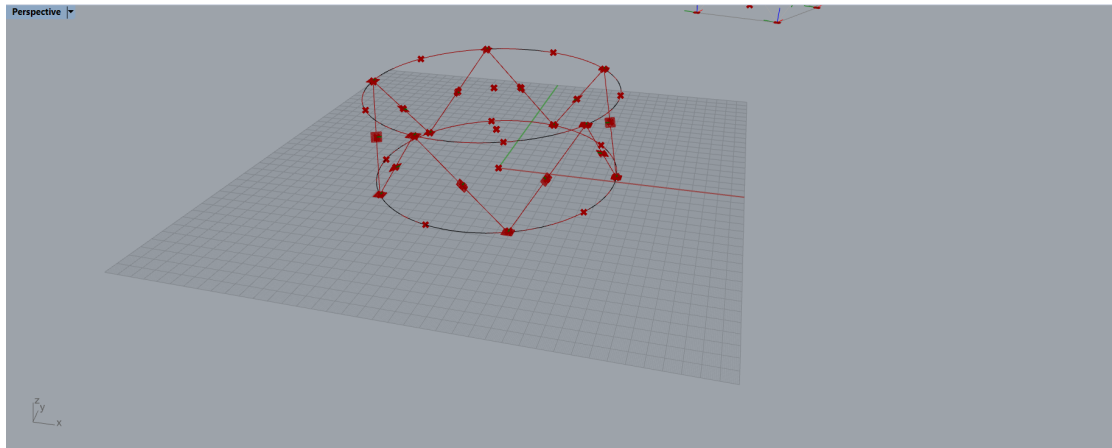


Figure 5.27 : Planes of the pentagon in the Rhino interface

In Figure 5.27 it can easily be seen how the top and bottom layers have horizontal planes which will be printed without any tilting of the extruder and how the middle sections of the sections of the polyline are tangent to the polyline, the extruder will have to bend and tilt to follow the orientation of the polyline until the middle point of the segment and then undo the bending when approaching the top or bottom planes.

Once that we have the desired planes we can tell the robot how to move through those planes. Firstly, we have to tell the order of printing, we use the command *merge* to order and to put join all the planes in a single command respecting the order. The bottom planes will go first, then the up and down segments of the polyline and finally the planes of the top pentagon. For the middle section we have to write the right order that the robotic arm will follow, therefore, a list item is used. If we were to live this alone maybe the robotic arm would start printing the polyline at the top part of the pentagon instead of the bottom and so on. Once we have all the planes in a single list as an output we use

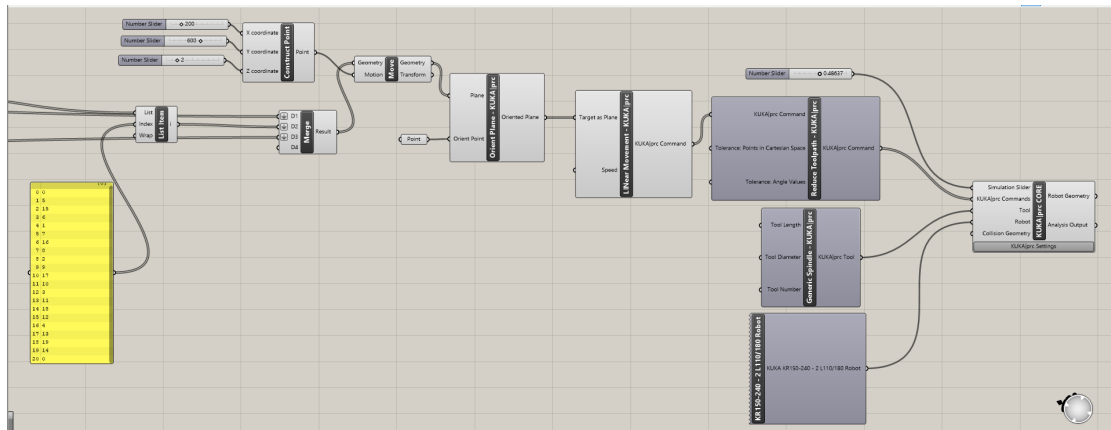


Figure 5.28 : KUKA|prc in Grasshopper interface

the command *move*, to move the object somewhere inside the printing table. This has also been done in the previous examples.

With the final geometry in the right place we can use the KUKA|prc plug in to instruct the movement of the KUKA arm. The code used for this is displayed in Figure 5.28. Firstly we use the command *OrientPlane-KUKA|prc* to orient all the x-axis of the planes towards a point, this is done to avoid collisions of the arm with the table or already printed parts of the structure. The output of this command gives us a list of oriented planes which will be used with the *Linear Movement- KUKA|prc* command. This command will move the robot in a straight line from one plane to the other.



Figure 5.29 : KUKA|prc in the Rhino interface

Next we will simplify the commands as much as possible using the *Reduce Toolpath-KUKA|prc* command always respecting the changes in orientation. This is used as an input for the *KUKA|prc CORE* command which includes all the core functionality from code generation to simulation of the movement. It includes a number slider that controls the simulation, you can insert the specific tool that it will be used or in my case use a generic tool. Moreover you can insert the specific KUKA robot that will be used to

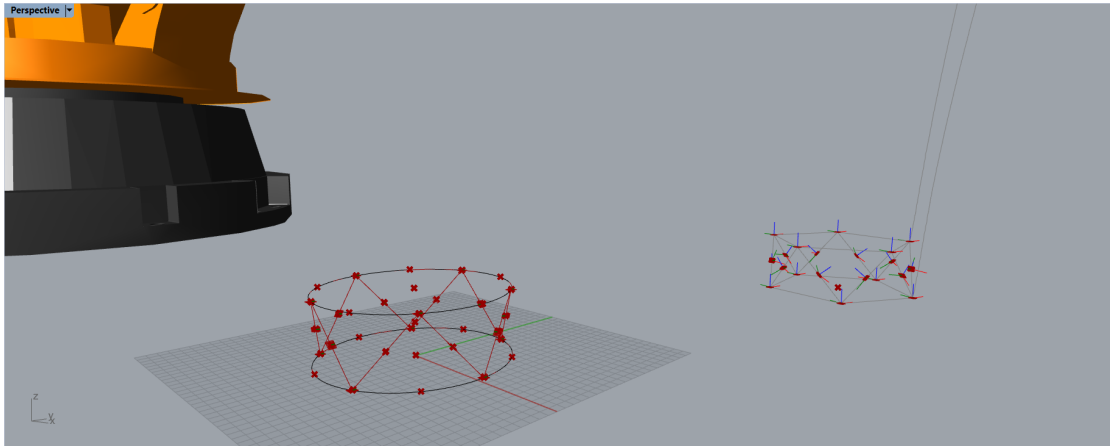


Figure 5.30 : KUKA|prc in the Rhino interface closer look

make sure that everything works correctly. Once all the inputs of the *KUKA|prc CORE* command are inserted we can see in the Rhino interface how a KUKA robotic arm appears, displayed in Figure 5.29, here it is seen the initial geometry, the one that has been moved to fit into the table and the point that was created to orient all the planes so we avoid collisions.

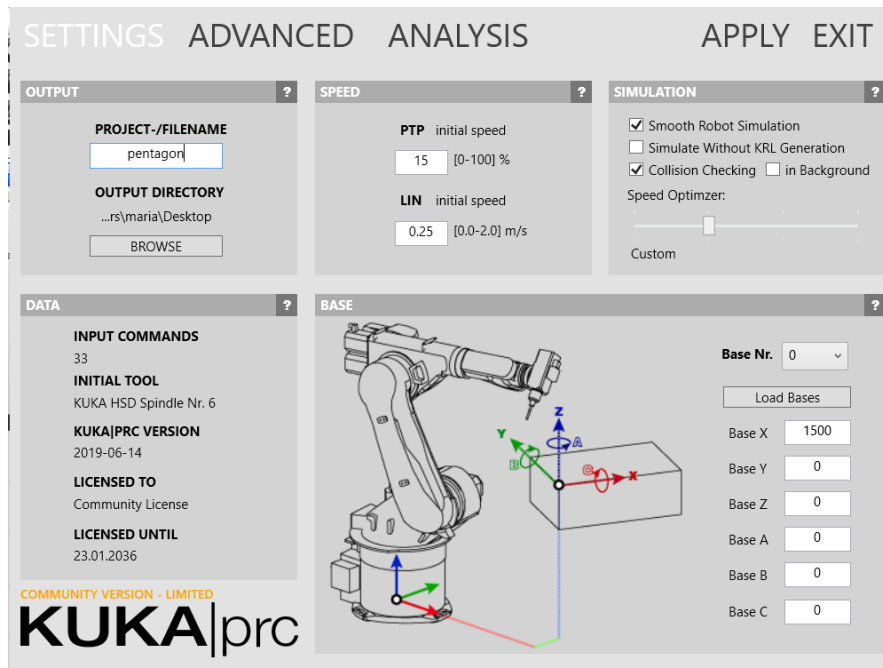


Figure 5.31 : KUKA|prc CORE

To see what happens in the *KUKA|prc CORE* in more detail we can enter the command and a new window appears seen in Figure 5.31. It is displayed how the filename of the project should be saved, the configuration of the speed, which right now is not important, the speed will be chosen later when printing. Also, the configuration of the base, where the printing bed will be placed, since it is a horizontal table the rotations of the base

A,B and C are zero, the Base X is 1500 mm since the table is placed at 1.5 m from the KUKA base. There are also some settings regarding the simulation, but they are not important for the actual printing.

If we click on the window analysis, the Figure 5.32 will appear. It shows in more detail the simulation of the movement of the robotic arm. On the horizontal axis of the graph the moment of the simulation is displayed in seconds, And on the vertical axis the position of each axis in each moment of the simulation is shown. If there were to be any collisions, it would be shown with a red vertical line the point where there is a collision.

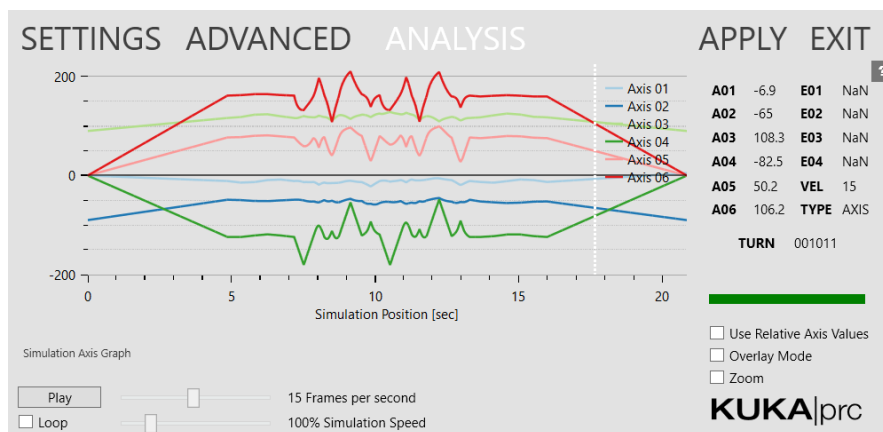


Figure 5.32 : Analysis of the movement of the KUKA robot

With the toolpath created and without any collisions visible, we can save the code generated by the KUKA|prc plug in. If we open it as an *.txt file, the commands seen in Figure 5.33 are displayed. The coordinates of the points are written in a cartesian coordinate system like the one shown in (5.1).

We have to remember that the code shown in Figure 5.33 corresponds to the movement of the robot arm, it does not include the extrusion of the material, and it is not a G-Code. In order to transform it into a G-Code we have to calculate the amount of filament that will be extruded for each point. Also we have to change the notation of the coordinates and include the G-values that will tell the extruder when to extrude filament or not. The calculation of the extrusion parameters is shown in Figure 5.34.

There are two different extrusion parameters, the ones for the top and bottom pentagons will be the same since the length of the sides of the pentagons is the same. However, the length of the segments of the middle polyline is not the same as the length of the top and bottom pentagons. To calculate the amount of filament that has to be extruded we first have to calculate the length of each segment. To do so, we use the command *length*, the input of which is the initial circumference, then we divide the total length by the number of points to calculate the length of each segment. In order to calculate the extrusion value for the G-Code equation (5.2) is used, its implementation in the grasshopper interface is displayed in Figure 6.38, *d* is the distance of each segment. We

```

;FOLD INI
;FOLD BASISTECH INI
GLOBAL INTERRUPT DECL 3 WHEN $STOPMESS==TRUE DO IR_STOPM ( )
INTERRUPT ON 3
BAS (#INITMOV,0 )
;ENDFOLD (BASISTECH INI)
;ENDFOLD (INI)

;FOLD STARTPOSITION - BASE IS 0, TOOL IS 6, SPEED IS 15%, POSITION IS A1 0,A2 -90,A3 90,A4 0,A5 0,A6 0,E1 100,E2 0,E3 0,E4 0
$BWDSTART = FALSE
PDAT_ACT = {VEL 15,ACC 100,APO_DIST 50}
FDAT_ACT = {TOOL_NO 6,BASE_NO 0,IPO_FRAME #BASE}
BAS (#PTP_PARAMS,15)
PTP {A1 0,A2 -90,A3 90,A4 0,A5 0,A6 0,E1 100,E2 0,E3 0,E4 0}
;ENDFOLD

;FOLD LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD
$VEL.CP=0.25
$ADVANCE=3
;ENDFOLD

PTP {E6POS: X 298.495, Y 607.774, Z 2, A 0, B 90, C -64.897, E1 0, E2 0, E3 0, E4 0, S 'B 010'} C_PTP
LIN {E6POS: X 298.495, Y 607.774, Z 2, A 0, B 90, C -64.897, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 222.643, Y 695.789, Z 2, A 0, B 90, C -68.156, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 115.495, Y 650.847, Z 2, A 0, B 90, C -70.859, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 125.127, Y 535.056, Z 2, A 0, B 90, C -69.209, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 238.227, Y 508.435, Z 2, A 0, B 90, C -65.305, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 298.495, Y 607.774, Z 2, A 0, B 90, C -64.897, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 298.495, Y 607.774, Z 2, A 0, B 90, C -64.897, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 286.682, Y 635.939, Z 36.941, A 112.533, B 41.171, C 104.379, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 274.868, Y 664.105, Z 71.882, A 0, B 90, C -66.305, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 248.755, Y 679.947, Z 36.941, A -31.383, B 41.148, C -126.284, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 222.643, Y 695.789, Z 2, A 0, B 90, C -68.156, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 192.205, Y 693.258, Z 36.941, A -175.261, B 41.163, C 135.496, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 161.768, Y 690.726, Z 71.882, A 0, B 90, C -69.88, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 138.631, Y 670.787, Z 36.941, A 40.927, B 41.151, C -98.332, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 115.495, Y 650.847, Z 2, A 0, B 90, C -70.859, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 108.497, Y 621.117, Z 36.941, A -102.995, B 41.191, C -175.275, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 101.5, Y 591.387, Z 71.882, A 0, B 90, C -70.604, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 113.313, Y 563.222, Z 36.941, A 112.961, B 41.16, C 106.779, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 125.127, Y 535.056, Z 2, A 0, B 90, C -69.209, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 151.24, Y 519.214, Z 36.941, A -31.102, B 41.143, C -129.149, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 177.352, Y 503.372, Z 71.882, A 0, B 90, C -67.177, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 207.79, Y 505.904, Z 36.941, A -175.265, B 41.126, C 138.936, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 238.227, Y 508.435, Z 2, A 0, B 90, C -65.305, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 261.364, Y 528.375, Z 36.941, A 40.526, B 41.177, C -100.64, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 284.5, Y 548.314, Z 71.882, A 0, B 90, C -64.46, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 291.498, Y 578.044, Z 36.941, A -103.493, B 41.153, C -171.063, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 298.495, Y 607.774, Z 2, A 0, B 90, C -64.897, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 274.868, Y 664.105, Z 71.882, A 0, B 90, C -66.305, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 161.768, Y 690.726, Z 71.882, A 0, B 90, C -69.88, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 101.5, Y 591.387, Z 71.882, A 0, B 90, C -70.604, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 177.352, Y 503.372, Z 71.882, A 0, B 90, C -67.177, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 284.5, Y 548.314, Z 71.882, A 0, B 90, C -64.46, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {E6POS: X 274.868, Y 664.105, Z 71.882, A 0, B 90, C -66.305, E1 0, E2 0, E3 0, E4 0} C_DIS
PTP {E6AXIS: A1 5, A2 -90, A3 90, A4 0, A5 0, A6 0, E1 100, E2 0, E3 0, E4 0} C_PTP
END

```

Figure 5.33 : pentagon.src

basically divide volume of the extruded material by the area of the contracted circle with a diameter of more or less 2.85 mm. The results of the pentagon segments and the polyline segments E values can be seen in Equations (5.3) and (5.4) respectively.

With the E values calculated we can modify the code shown in Figure 5.33 and transform it into G-Code as shown in Figure 5.35. This was done manually since I could not find a way to modify the KUKA code and transform it into G-Code within the grasshopper interface. Since it is a relatively simple figure, with not many commands it was easy to do. In Section 2.2 it is explained how to generate a G-Code and what commands must be included, also the G-Codes written by the Silkorm generator for the simple cube and the horizontal panel were very helpful to see the overall structure and check that I did not miss any commands. It is important to think when the robotic arm will be extruding filament and when it won't. For example, for the first coordinate it is not

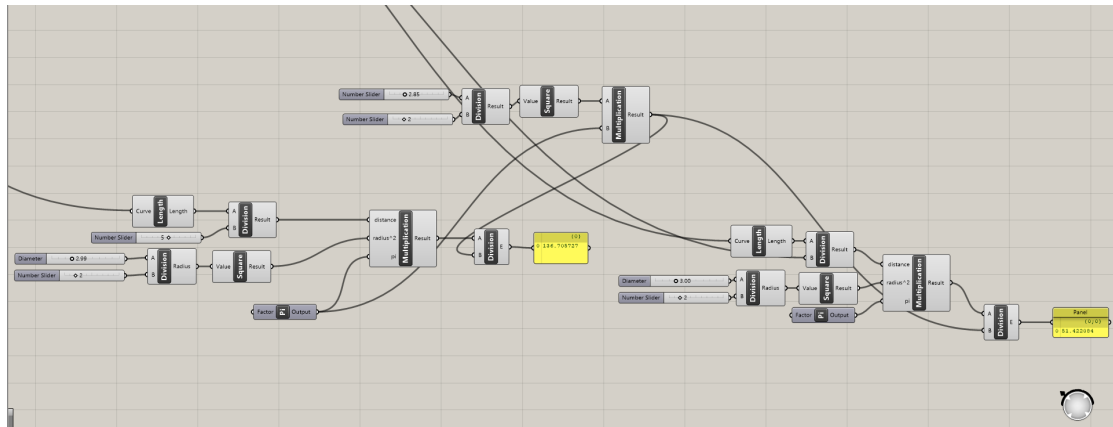


Figure 5.34 : Extrusion value

extruding any material but in order to get to the second one it will extrude material and create the pentagon. The final code is displayed in Figure 5.35, there it can be seen how there are only rotations for the x and y axis, we only have A and B values of rotation. This is because if we add the rotation in the z axis (C value) we could end up twisting the cables that are connected to the robotic arm or even breaking them. We don't know if the robotic arm will be able to read this code or if the C value is necessary.

$$E = \frac{\left(\frac{3}{2}\right)^2 * \pi * d}{\left(\frac{2.85}{2}\right)^2 * \pi} \quad (5.2)$$

$$E_{pentagon} = \frac{\left(\frac{3}{2}\right)^2 * \pi * d}{\left(\frac{2.85}{2}\right)^2 * \pi} = 136.7mm \quad (5.3)$$

$$E_{polyline} = \frac{\left(\frac{3}{2}\right)^2 * \pi * d}{\left(\frac{2.85}{2}\right)^2 * \pi} = 51,42mm \quad (5.4)$$

Once the G-Code has been created, the only thing left to do is to check that this code works correctly in our printer.

To check this we open RobotDK and upload the code file. Right away we get an error that tells us that the part cannot be printed because the angles are not correct. After some thinking and trying we discovered that the axis of the toolhead of the robotic arm were not in the same orientation as the axis of the toolhead of Grasshopper. In order to solve this problem we had to change all the B values, perform a rotation of -90° . With all the B values rotated we load the file once again in RobotDK and we don't get any errors, the figure is ready to print.

```

M83 ; use relative distances for extrusion
G90 ; use absolute coordinates
G21 ; set units to millimeters
G92 E0 ; reset extrusion distance
M104 S280 ; set temperature
M109 S280 ; wait for temperature to be reached
G1 X-200 Y200 Z2 F1200 E1
G92 E0

G1 F1200 X298.495 Y607.774 Z2 A0 B90
G1 F1200 X222.643 Y695.789 Z2 A0 B90 E136.7
G1 F1200 X115.495 Y650.847 Z2 A0 B90 E136.7
G1 F1200 X125.127 Y535.056 Z2 A0 B90 E136.7
G1 F1200 X238.227 Y508.435 Z2 A0 B90 E136.7
G1 F1200 X298.495 Y607.774 Z2 A0 B90 E136.7

G1 F1200 X286.682 Y635.939 Z36.941 A112.533 B41.171 E51.42
G1 F1200 X274.868 Y664.105 Z71.882 A0 B90 E51.42
G1 F1200 X248.755 Y679.947 Z36.941 A-31.383 B41.1485 E51.42
G1 F1200 X222.643 Y695.789 Z2 A0 B90 E51.42
G1 F1200 X192.205 Y693.258 Z36.941 A-175.261 B41.163 E51.42
G1 F1200 X161.768 Y690.726 Z71.882 A0 B90 E51.42
G1 F1200 X138.631 Y670.787 Z36.941 A40.927 B41.151 E51.42
G1 F1200 X115.495 Y650.847 Z2 A0 B90 E51.42
G1 F1200 X108.497 Y621.117 Z36.941 A-102.995 B41.191 E51.42
G1 F1200 X101.5 Y591.387 Z71.882 A0 B90 E51.42
G1 F1200 X113.313 Y563.222 Z36.941 A112.961 B41.16 E51.42
G1 F1200 X125.127 Y535.056 Z2 A0 B90 E51.42
G1 F1200 X151.24 Y519.214 Z36.941 A-31.102 B41.143 E51.42
G1 F1200 X177.352 Y503.372 Z71.882 A0 B90 E51.42
G1 F1200 X207.79 Y505.904 Z36.941 A-175.265 B41.126 E51.42
G1 F1200 X238.227 Y508.435 Z2 A0 B90 E51.42
G1 F1200 X261.364 Y528.375 Z36.941 A40.526 B41.177 E51.42
G1 F1200 X284.5 Y548.314 Z71.882 A0 B90 E51.42
G1 F1200 X291.498 Y578.044 Z36.941 A-103.493 B41.153 E51.42
G1 F1200 X298.495 Y607.774 Z2 A0 B90 E51.42

G1 Z75
G1 F1200 X284.5 Y548.314 Z71.882 A0 B90
G1 F1200 X274.868 Y664.105 Z71.882 A0 B90 E136.7
G1 F1200 X161.768 Y690.726 Z71.882 A0 B90 E136.7
G1 F1200 X101.5 Y591.387 Z71.882 A0 B90 E136.7
G1 F1200 X177.352 Y503.372 Z71.882 A0 B90 E136.7
G1 F1200 X284.5 Y548.314 Z71.882 A0 B90 E136.7

G1 Z250
G92 E0 ; reset extrusion distance
M104 S0 ; turn off temperature
M84 ; disable motors

```

Figure 5.35 : G-Code pentagon

6 Validation

Once the code for the lattice structures has been created, we can start printing the parts and prove whether this method is possible or not. When using the printer there are some aspects that should be taken care of:

- **Collision avoidance:** When printing the part, the robotic arm cannot collide with already printed parts. In principle, this has been checked when we uploaded the codes created into RobotDK and we made sure that everything worked fine.
- **Right cooling temperature:** Cooling is crucial for free form printing, with the right cooling the filament that is being extruded will be able to cool off quickly and stick properly to the table.
- **Proper adhesion:** it is important that the upper layers that are being printed stick properly to the lower layers. For example when printing the horizontal panel the top part should stick properly with the middle part.

The workflow to follow for the printing of the parts is the same for every case. We load the file into the RobotDK interface, we check that no errors appear, then we run the preprocessor and finally we run the code that is sent to the robotic arm via ethernet. The filament is loaded to the extruder through an air pump that has to be activated manually when needed.

Inside RobotDK we have a few python scripts that are needed for the correct functionality of the printer, the already mentioned preprocessor script that is needed to check that everything works fine. Another important script is the SetExtSpeed that will be modified in some cases to make sure that the filament being extruded and the movement of the arm work simultaneously.

Before starting to print it is crucial to calibrate the KUKA robot, this is done manually following a series of instructions. Also, a heating calibration is done, already mentioned in Section 3 to check that the sensors are correctly placed and that they are heating the three zones adequately.

Now that the robotic arm is ready we send a series of commands manually to check that the nozzle is correctly extruding the filament. When we first did this, the filament got stuck in the pulsar extruder and it solidified there. Therefore, no material was coming out. As a result the nozzle was full of solidified material and we had to leave it in the oven overnight to be able to take it out. Once this small incident was solved, we tried this procedure again and everything worked as planned.

With all these aspects in mind we can start printing. This chapter will be divided into three subsections each one focused on the printing of the corresponding code that we created in Section 5.3.

6.1 Simple cube

In this section the results of the printing of the simple cube will be displayed. Since it is a very simple structure many trials were done in order to find the right conditions for the printing.

6.1.1 Cube N°1

As it was explained in Chapter 3 the pulsar extruder used in the KUKA robot has a tri-zone heating control, we have sensors at the zone where the filament enters the extruder, at the middle, the so called compression zone and one final sensor placed at the nozzle. These sensors are responsible for correctly heating those parts. The temperatures of each can easily be changed thanks to an available interface. Throughout the validation of the project, those three values will be modified to find the correct settings. The values chosen for the first print are seen in Table 6.1. The temperature of the printing bed is also shown there.

In order to avoid collisions which is one of the main concerns we decided to make the cube a little bigger, initially it was 75mmx75mmx75mm, we went into the Grasshopper file and since it is fully parametric we doubled the distances making each side of 150 mm.

	temperature (°C)
entry zone	205
compression zone	210
nozzle	175
bed	130

Table 6.1 : Temperature data of Cube N°1

With the values show in Table 6.1 selected we run the preprocessor script and then the code `*cube_final`. The KUKA arm moves to the initial position and starts printing. In Figure 6.1 it can be seen how the figure is being printed. It is clear that the vertical struts are not correctly solidifying, some parts of it are more or less vertical but the overall structure falls down.

Looking at the Figure 6.1 it can be seen that the part is well stuck to the bed, which means that the bed temperature was well chosen, maybe for the next try we could try a bed a little colder to see if the material also sticks to the table.

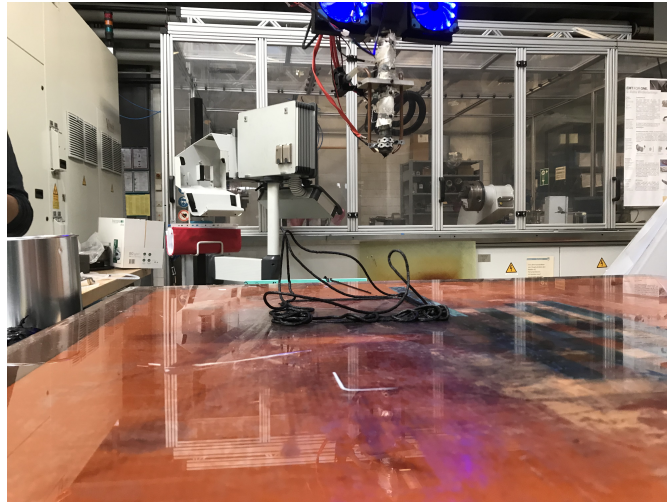


Figure 6.1 : Cube N°1

Overall, I think that the material coming out of the nozzle was very liquid and it did not have enough time to solidify. Therefore, in the next prints the temperature of the sensors should be lower, specially at the compression zone, where the material melts. Also, we could lower the speed of the KUKA so it takes longer to print and has more time to solidify.



Figure 6.2 : Cube N°1 top view

In Figure 6.2 a top view of the final cube can be seen, some of the vertical struts that were in an upright position in Figure 6.1 have fallen down, since they were not solidify quickly enough. Therefore, the cooling was not working properly, checking the configuration again we found out that the cooling system that I had design was not turned on.

Even though that the print does not look like a cube at all, in Figure 6.2 it can be appreciated how the print does have a square form which is pretty good. Also, the commands that the robotic arm followed were like the ones I was expecting, therefore, the code has been well written. The only thing remaining is to get the right values for the extrusion speed, the speed of the robotic arm and the temperature.

6.1.2 Cube N°2

For the printing of the second cube we have done some temperature adjustments seen in Table 6.2, we have decreased the temperature of the entry zone and the compression zone to have a not so liquid material, and we have decreased the bed temperature to check if it still sticks to the table.

	temperature (°C)
entry zone	195
compression zone	205
nozzle	180
bed	100

Table 6.2 : Temperature data of Cube N°2

In addition to the temperature data we have also modified some settings which are seen in Table 6.3. We have reduce the speed factor of the robotic arm to 60% so the arm moves slower and the material is able to cool down. We have doubled the amount of material that is being extruded so it has more consistency and is able to stay in its position. We have also turned on the fans, this will help to quickly cool down the part.

speed factor	60%
extrusion factor	200%
fans	on

Table 6.3 : Cube N°2 settings

The part being printed can be seen in Figure 6.3. It can be seen how the vertical struts of the cubes stay more or less in the vertical position, and the diagonal part is also visible, it is not perfect but it's a start. Nevertheless, we are still having problems with the cooling, since the part is a little wobbly. One problem could be that the fans are not powerful enough for the cooling. When trying the cooling system in Section 3 it was said that the system did not exhaust as much air as we were expecting, maybe for future modifications this should be changed and look into other cooling options.

One thing that we didn't consider when developing the code is the effect of gravity. In Figure 6.3 the vertical strut that can be seen is shorter than 150 mm that is the total height. Also, since gravity pulls down the part, when the arm is printing the top part of the cube, it does not stick with the vertical struts. This is something that it should be looked into.

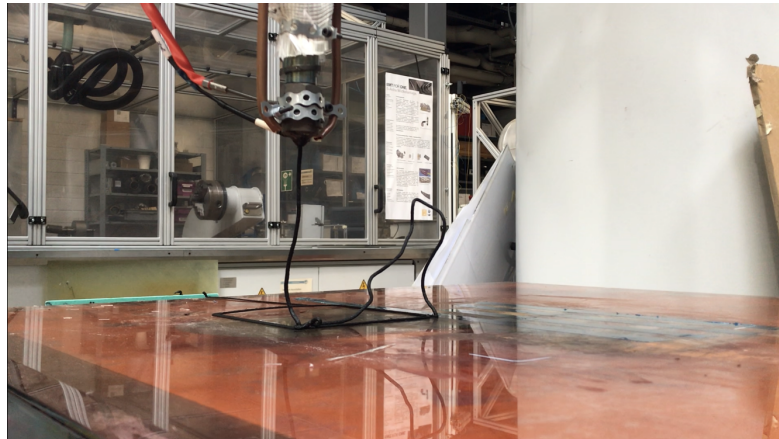


Figure 6.3 : Printing of Cube N°2

We are having problems with the synchronization of the robot and the extruder. In Figure 6.4 it can be seen how the nozzle has extruded more material than necessary, there last edge of the top of the cube goes upwards and makes some kind of a twist. There should not be any material there, the last edge of the top should stay horizontal. It looks like the nozzle keeps extruding material when it moves away when it has finished printing the cube.



Figure 6.4 : Cube N°2

Overall, the result has greatly improved if we compare it with the first print seen in Figure 6.2. Now the print almost looks like a cube, some the vertical struts are well

defined, even the diagonals are almost well printed. In Figure 6.4 it can be seen an almost perfect diagonal at the back which is a very good sign.

6.1.3 Cube N°3

For the third try we keep decreasing the values of the temperature of the entry and compression zones so the filament does not come out very liquefied and is able to solidify faster, the values chosen can be seen in Table 6.4.

	temperature (°C)
entry zone	175
compression zone	195
nozzle	180
bed	100

Table 6.4 : Temperature data of Cube N°3

Regarding the settings seen in Table 6.6, the speed factor has two values. To start printing we will use the 60% of the robotic arm speed but then when we start going up to print the vertical struts we reduce even more the speed to 45% so it has time to solidify, also the fans are off. Let's see if this configuration works better than the previous.

speed factor	60% and 45%
extrusion factor	200%
fans	off

Table 6.5 : Cube N°3 settings

Figure 6.5 shows the third cube being printed. Here it can be seen how we still have problems of synchronization between the extruder and the robotic arm since the KUKA is already moving away from the part and the nozzle keeps extruding filament. Also, it seems that the vertical struts were more vertical and better cooled down in the previous print, shown in section 6.1.2, this could be because in this print the fans were off. Therefore, even though that the cooling system is not very powerful there is a significant difference between having them on and turning them off.

Having decreased the speed of the robotic arm to a lower value when printing the vertical struts and the top part. Instead of pushing the material out of the nozzle we are pulling the material out of the nozzle which gives the material more time to solidify. In Figure 6.6 it can be seen how the top part looks better than in previous prints, but the vertical and diagonal struts were better in the previous case.

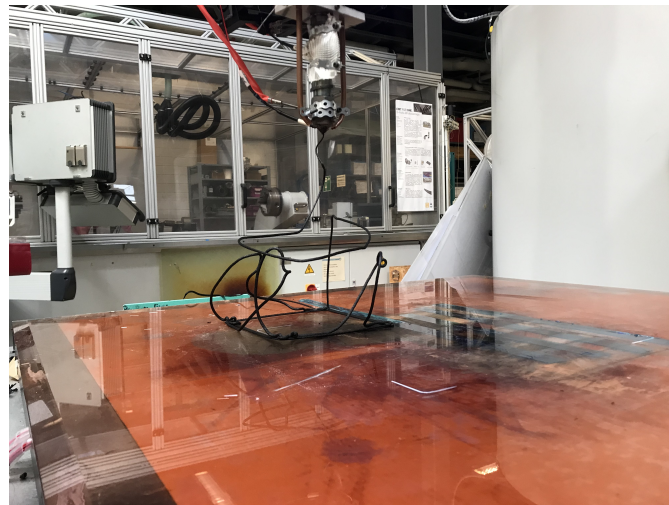


Figure 6.5 : Printing of Cube N°3

In general, the material might be one of the main issues for the cooling problem that we are having. As it was said in Section 4, we are using PP, which has a low glass temperature, this means that it takes longer to cool down, with a material with a higher glass temperature such as ABS we could have achieved better results since the material extruded would have solidify much faster.



Figure 6.6 : Cube N°3

6.1.4 Cube N°4

For the fourth try we decreased a little the value of the nozzle temperature to check if we achieved better results. For the entry and compression zones the previous values were maintained, this can be seen in Table 6.6.

In Section 6.1.3 we tried printing with two different speed factors, now we will try printing the whole structure with the smaller value and see if the overall result gets

	temperature (°C)
entry zone	175
compression zone	195
nozzle	175
bed	100

Table 6.6 : Temperature data of Cube N°4

better, specially the vertical and diagonal struts that in the previous print were quite disappointing. Also we decreased the extrusion factor to 180% to try to synchronize the movement of the robotic arm and the extrusion of the filament. We turn the fans on again since in the previous print the cooling of the vertical struts was not as we expected. All this is shown in Table 6.7.

speed factor	45%
extrusion factor	180%
fans	on

Table 6.7 : Cube N°4 settings

Figure 6.7 shows how Cube N°4 is being printed. The result is much better than in previous tries. Having printed everything at a much slower rate and having turned the fans on has allowed the filament to cool correctly and achieve almost vertical edges. Also the diagonals are looking very good. Nevertheless, if we look closely at the base of Figure 6.7 it can be seen how the bottom part of the cube is not fully attached to the printing bed.



Figure 6.7 : Printing of Cube N°4

The problem with the printing not sticking to the bed properly could imply further problems. If the print does not stick to the bed the base could move mid way through printing. This means that when printing the diagonals the extruder will not find the base at the bottom and it could collide into the table, which could break it. It is important to solve this for future prints.



Figure 6.8 : Cube N°4

One of the main concerns when printing the lattice structures was that the layers stuck properly to previous layers. In Figure 6.8 the final print of Cube N°4 can be seen. Here the vertical struts and the diagonal lines are well attached to the bottom printed layer, the base of the cube. However, the top square that closes the cube does not stick to the vertical struts. There are many reasons for this, the main one being gravity, since the vertical struts are pulled down due to the gravity effect, when the top square is being printed at the desired height it does not find anything to get stuck to. Maybe if we try printing everything slower, the vertical struts will solidify better and they won't get pulled down as much.

6.1.5 Cube N°5

We were happy with the temperatures chosen for the last print, therefore, for the fifth cube we kept the same ones. However, in order to try and solve the sticking to the table problem we increased the temperature of the bed to 110°, seen in Table 6.8. Since the printing bed will be hotter, the filament printed on the table we hope that it will attach better.

The other printing settings can be seen in Table 6.9, the extrusion factor has been kept the same that the one in the previous print, but we have decreased a lot the speed factor. For this print we will try a 15% speed factor, to check if by printing very very slowly the material sticks and solidifies better. Also, it can be seen that the fans will be both off

	temperature (°C)
entry zone	175
compression zone	195
nozzle	175
bed	110

Table 6.8 : Temperature data of Cube N°5

and on. For the printing of the base square of the cube the fans will remain off, to let the print stick properly to the bed. However, when starting to print the vertical and diagonal struts, we will turn the fans to 100% so they help cool down the part.

speed factor	15%
extrusion factor	180%
fans	off/on

Table 6.9 : Cube N°5 settings

We had high hopes for this configuration, but the results were not very successful. In Figure 6.9, the Cube N°5 is being printed. It can be seen how we had to use tape to stick the print to the table so it would not move. Even though that we turned off the fans for the bottom layer and the printing bed temperature was increased. I think that the main problem was the speed of the robotic arm, it was too slow to print, for the next printings this will be definitely changed.

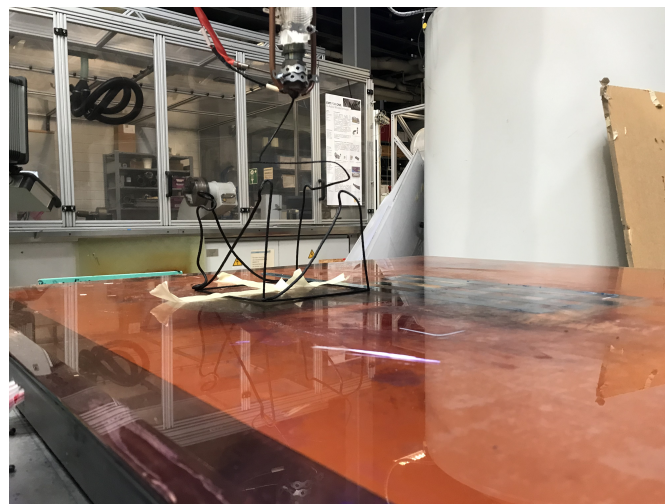


Figure 6.9 : Printing of Cube N°5

In Figure 6.10 the final print can be seen. It is easy to see how bottom layer is not parallel to the floor due to the print not sticking to the bed. However, the vertical struts

are looking good, having turned on the fans for this part has been beneficial. When printing the diagonals the filament does a strange curve, this might be because of the slow speed we were printing with. Another big disadvantage of printing with low speed is the time that it takes to print is larger, for such a simple structure it might not be a big deal but for larger parts it could be an issue.



Figure 6.10 : Cube N°5

When the printing was done and the robotic arm moved away it took part of the top layer with it, this is why it cannot be seen in Figure 6.10, we have to find a way to cut the extrusion of filament when finishing the printing.

6.1.6 Cube N°6

From this print on we decided to change the size of the cube to its original size: 75 mm x75 mmx75mm, we only had to go into the Grasshopper file and change the parametric slider to the desired length. The main reason behind this change was that making it smaller would improve the printing, specially the diagonal and vertical struts, also since there were no collisions with the big print between the extruder and the printed filament, we wanted to try it with a smaller size.

	temperature (°C)
entry zone	175
compression zone	195
nozzle	175
bed	110

Table 6.10 : Temperature data of Cube N°6

The temperature settings shown in Table 6.10 are exactly the same as with the previous print. However the other printing settings seen in Table 6.11 have been modified, we are no longer printing at such low speed, but at 45% which in Section 6.1.4 worked well. For the vertical and diagonal struts the speed will be reduced to 20%. Also, we still have the same configuration for the fans, they will remain off for the printing of the bottom layer, allowing the material to stick properly to the bed and when printing the vertical and diagonal parts we will turn it on.

speed factor	45% and 20%
extrusion factor	180%
fans	off/on

Table 6.11 : Cube N°6 settings

In Figure 6.11 it can be seen the final print of the Cube N°6. As expected is much smaller since the length has been changed. Also, the bottom layer attached better to the printing bed which is better than in the previous prints, but we still get some warping and tape was used to make sure it did not move. However, the vertical and diagonal struts did not print well. It can be seen in Figures 6.11 and 6.12 that one of the diagonal struts broke.



Figure 6.11 : Cube N°6 front view

We are still having problems with the attachment of the top layer, it solidifies well but since the vertical struts are pulled down due to gravity, the top part does not stick with the already printed struts. I thought that by decreasing the speed when printing the vertical and diagonal sides the effect due to gravity would be diminished but it has not worked.

Moreover, we are still having synchronization problems between the robotic arm and the extruder, in Figure 6.11 it can be seen how the nozzle has kept extruding material when

the robotic arm was moving away from the part, since we have that vertical filament at the top of the cube.



Figure 6.12 : Cube N°6 side view

Overall, it seems like this one has been one of the worst prints yet, however, we have to take into account that we have changed the size of the cube and the whole configuration should change with it. This might take us a few more prints to accomplish.

6.1.7 Cube N°7

One of the main concerns that we have been talking about is that the top part of the cube is not sticking to the vertical struts. It has been previously said that this might be because of gravity, that pulls down the vertical struts, and when the top layer is being printed it does not find the vertical strut to stick with. In order to reduce this effect, we tried slowing down the the robotic arm when printing the vertical struts so it had time to cool down properly. This was done in several trials, shown in Sections 6.1.5 and 6.1.6 and it did not work, therefore, we must come up with another solution.

	temperature (°C)
entry zone	175
compression zone	210
nozzle	175
bed	110

Table 6.12 : Temperature data of Cube N°7

To make sure that the top layer and the vertical struts were at the same height we decided to modify the code. We changed the height of the top layer but maintained the initial

height of the vertical struts. By doing this, the vertical struts will be pulled down by gravity but now the top layer will print at a lower height therefore colliding and sticking with the vertical struts. To find how much we had to decrease the height of the top layer we measured how long were the vertical struts in the previous print. An offset of 5mm was considered a good first try. Therefore, the vertical struts will be printed at 75 mm but the top layer will be done at 70 mm. Since this was a simple code, with few commands, it was modified manually, but it could also be done in the Grasshopper interface.

speed factor	35%
extrusion factor	100%
fans	off/on
offset	5 mm

Table 6.13 : Cube N°7 settings

The temperature settings chosen for this print can be seen in Table 6.12, and the factors considered are shown in Table 6.13. For this print we decided to change the speed factor to 30% throughout the whole print, and keep the extrusion factor to 100% in other words, we use the extrusion factor that was calculated by the Silkworm plug in. Like in the previous prints, we will turn off the fans for the base square and turn them on for the rest of the printing.



Figure 6.13 : Front view of Cube N°7

Figure 6.13 the final print of Cube N°7 can be seen. We finally see how the top layer collides with the vertical struts, which means that our idea of changing the height of the code worked. Also, the extruder speed and the robotic arm speed are synchronised, we don't have any extra filament hanging. However, it can be seen in Figures 6.13 and 6.14 that the structure tilts a little to the left, maybe we should print this parts slower, to let

the material cool down , or change the cooling system to one more powerful. It is also important to notice that the part does not move at the printing bed, this means that the bed temperature and the decision of turning off the fans in this section was well done.



Figure 6.14 : Side view of Cube N°7

Now that we have a more or less good result, the only thing remaining would be to vary a bit the configuration to see how the printing varies, and whether better prints can be achieved.

6.1.8 Cube N°8

For the Cube N°8 we kept the same temperature conditions than in Section 6.1.7 seen in Table 6.14 since in the previous print they worked pretty well. In Table 6.15 the other configuration factors can be seen, we have further increased the offset to see if we can achieve a greater improvement, and we can finally have a collision between the vertical edges and the top square. The rest of the factors are maintained since they worked very well in the last print, also we keep turning off and on the fans when needed.

	temperature (°C)
entry zone	175
compression zone	210
nozzle	175
bed	110

Table 6.14 : Temperature data of Cube N°8

The results are shown in Figures 6.15 and 6.16. Firstly, it must be said that once again, we have achieved a good bed adhesion since the print did not move throughout the printing. Also, one of the top edges did collide with a vertical strut, it can be seen in

speed factor	35%
extrusion factor	100%
fans	off/on
offset	10 mm

Table 6.15 : Cube N°8 settings

Figure 6.16, but for the rest of edges we are not able to collide. When printing the vertical struts, the collision with the bottom edges was perfect. In this print we have also achieved the synchronizing of the robotic arm and the extruder since no more material is coming out once it has finished printing.



Figure 6.15 : Front view of Cube N°8

However, for this print, the vertical and diagonal sections were not as vertical and straight as in the previous Section 6.1.7. It seems like there was too much material coming out of the nozzle. This is strange since, the speed and extrusion factors were not changed from those in Section 6.1.7 so the amount of filament extruded should be the same in both cases. One possible answer to this issue could be the amount of filament that there was available. Maybe in Section 6.1.7 there was little filament left and when trying to print the Cube N°7 we loaded the entry zone with filament by activating the air pump and hence there was a lot more material to be extruded. In future cases we should try and decrease the extrusion speed to less steps per mm and make sure that there is enough material available.

It is interesting to see how even though that the only thing we changed from the previous print was the offset of the top layer the final print looks very different from the one in Section 6.1.7. This shows us how there are some factors that we cannot change and they still interfere with the results, aspects like the amount of material inside the entry

chamber or maybe the temperature of the environment can make the print more wobbly at some parts and that we cannot change.



Figure 6.16 : Side view of Cube N°8

6.1.9 Cube N°9

For the printing of Cube N°9 we have changed a bit the temperature settings of the printing, they are shown in Table 6.16, from Section 6.1.8 the only value that has been changed is the temperature of the inlet which has been decreased. We have done this to try to make the printing less wobbly than the previous case.

	temperature (°C)
entry zone	165
compression zone	210
nozzle	175
bed	110

Table 6.16 : Temperature data of Cube N°9

When looking at Table 6.17 it can be seen how we have also changed some values. We are going to print at two different speed factors, for the horizontal base we will be using 35% and for the rest of the structure we will decrease to 30%, let's see if now we get more consistent vertical struts. We keep the same offset configuration since in Section 6.1.8 worked well. We still turn on and off manually the fans and we keep the same extrusion factor.

The resulting print can be seen in Figures 6.17 and 6.18. First of all when printing this cube we had some adhesion problems, the print was not sticking properly to the printing bed and we had to use tape to fix it to the table. However, when we added the tape it was

speed factor	35% and 30%
extrusion factor	100%
fans	off/on
offset	10 mm

Table 6.17 : Cube N°9 settings

too late since the base had already moved. In Figure 6.17 can be seen how the vertical strut at the back of the image has been printed in mid air. I don't know why having kept the same exhaust and bed temperature as in Section 6.1.8, the printer did not stick to the table this time.



Figure 6.17 : Front view of Cube N°9

Overall, the vertical and diagonal struts look much better than in Section 6.1.8, which means that having decreased the speed factor in these parts has been a good idea. However, some of the diagonals are not cooling down as well as expected, they are not fully straight, this is because we are having collisions with the pipes of the cooling system that are placed next to the nozzle of the extruder and the already printed diagonal part, which since it is not fully cooled down it moves making a strange curve instead of a straight diagonal. When printing the bigger cube 150mm x 150mm x 150 mm we did not encounter any collisions, everything was bigger and spread apart. Now, with smaller lengths, the extruder has to do stepper movements and collision is possible.

When loading the file in RobotDK, there is an option to run the simulation and check that no collisions take place, this is done before each print but the simulation does not take into account the cooling system that we have designed and inserted into the pulsar extruder. For future improvements, the pipes of the cooling system could be placed



Figure 6.18 : Side view of Cube N°9

somewhere else, leaving the nozzle free to move. Once again, in this print we have also achieved a good synchronization between the KUKA arm and the extruder.

I think that this print has been one of the best ones yet, despite the fact that we had some collisions and that the print wouldn't stick to the table. However, Figure 6.18 shows an almost perfect cube that we were looking for.

6.1.10 Cube N°10

Before printing Cube N°10 and Cube N°11 we decided to try the horizontal panel which is explained in Section 6.2. with the feedback we received from those prints we tried again the simple cube since it is an easy and fast structure to print.

	temperature (°C)
entry zone	195
compression zone	210
nozzle	175
bed	110

Table 6.18 : Temperature data of Cube N°10

The temperature settings that we choose are shown in Table 6.18, which are the same ones that we used for the last printing of the horizontal panel. The temperature of the entry zone is increased with respect to previous prints but the rest remains the same.

The rest of the settings chosen can be seen in Table 6.19. They are the same we used for the horizontal panel, and very similar to those in sections 6.1.8 and 6.1.9. We will start printing at the correct speed the base of the cube but then decrease the printing speed

speed factor	100% and 30%
extrusion factor	100%
fans	off/on
offset	10 mm

Table 6.19 : Cube N°10 settings

to 30% so the vertical struts have enough time to cool down. We keep using the on and off configuration of the fans to allow printing adhesion. Even though that in previous sections like 6.1.8 and 6.1.9 it did not work, I think that if the fans were on for the whole print the adhesion would be even worse.



Figure 6.19 : Front view of Cube N°10

The final result of this print can be seen in Figure 6.19. The vertical struts are almost perfect, which means that the speed configuration was correct. It can be seen that the back diagonal is broken due to the collision of the cooling system with already printed parts, in this diagonal the extruder is going downwards. This can only be improved by changing the dimensions of the cube, instead of having a cube making a rectangle with a greater angle between the vertical and the diagonal. However, when the extruder is going upwards and printing another diagonal the result is almost perfect, this can be seen in Figure 6.20. Therefore, we are only having collision problems when the extruder goes downwards in steep angles.

Once again we have not achieved a synchronization between the extruded filament and the movement of the robot since there is a vertical line of filament going upwards as the robotic arm moves away. I don't know why this happened since the settings used are very similar to the ones used in Section 6.1.9 and we did not have any problems there. I

think that the problem might be in the amount of filament that there was available in the extruder.

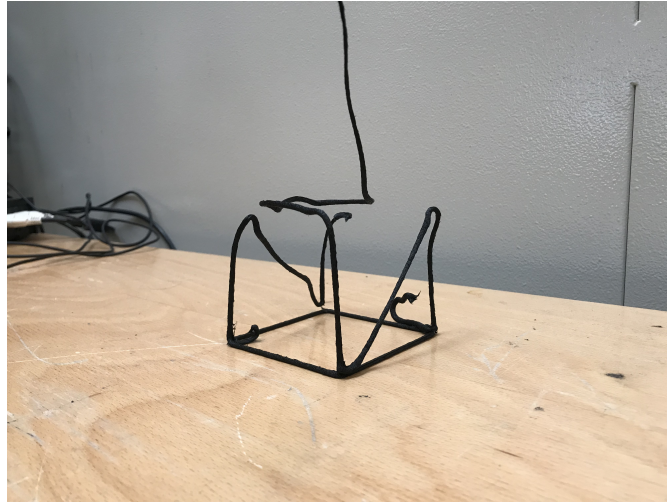


Figure 6.20 : Side view of Cube N°10

6.1.11 Cube N°11

Cube N°11 is the last cube we printed for the proof of concept of this first code. The temperature settings used are shown in Table 6.20, they are the same of the previous section 6.1.10, since they worked very well, specially with the bed adhesion, the printing table and the temperature of the extruder were okay to ensure that the print did not move from the table.

	temperature (°C)
entry zone	195
compression zone	210
nozzle	175
bed	110

Table 6.20 : Temperature data of Cube N°11

The same factor settings that in Section 6.1.10 are used for this print, shown in Table 6.21. There we see a new command called smooth. The functionality of this command is better explained in Section 6.2 since we first used it when printing the horizontal panel. Turning the smooth command off allows us to achieve the synchronization between the filament being extruded and the movement of the robotic arm, what we had been trying for several prints and we could not achieve. The smooth command is normally used in normal 3D printing to make cure and flatten the printing parts, for our 3 axis printing is no longer needed.

speed factor	100% and 30%
extrusion factor	100%
fans	off/on
offset	10 mm
smooth	off

Table 6.21 : Cube N°11 settings

Figure 6.21 shows how Cube N°11 looks like. Finally, the filament stops right where it is supposed to and the nozzle does not extrude more filament when the arm is moving away. This is thanks to the removal of the command smooth from the RobotDK code. In this print we have not had any collisions with the cooling system and all edges are looking more or less straight.



Figure 6.21 : Front view of Cube N°11

Even though that the top square of the cube is fully printed, we have not been able to make it stick to the vertical struts. The offset that we placed for the top part is more or less adequate, maybe it should be a bit decreased. When we were printing the top square and the extruder was close to the vertical struts, the material from the vertical struts did not melt and bond with the filament that was coming out of the nozzle, therefore, no adhesion was possible. We have been using PP as the filament material, which is difficult to adhere to the bed and due to its semi-crystalline nature, very prone to warping. Maybe for future improvements we could try with a material that has excellent layer bonding like PLA.



Figure 6.22 : Side view of Cube N°11

6.2 Horizontal lattice structure

Once we had tried a few times printing the single cube and more or less achieved the desired result we decided to print the second code, the horizontal panel, which is made of unit cells similar to the single cube already printed. This print is much bigger, hence it will take longer to print. Having a bigger print will allow us to change the configuration mid way through printing and see the differences that changing the setting makes in the same print.

The workflow to print the horizontal panel is the same that for the single cube. The G-Code that we created in the Grasshopper interface is loaded into the RobotDK interface, after checking that no collision take place and that the print is inside the printing bed we can start to print.

6.2.1 Horizontal panel N°1

The temperatures used for this print are shown in Table 6.22, the bed temperature has been increased to try and stick the material better to the printing table. Since it is the first print we did with this code the temperature of the entry and compression zones was also increased to make sure that the material was flowing properly. When loading the file into the RobotDK interface and looking at the displayed code we saw that some speed values were too large, and that it might be dangerous to move the robotic arm at such large speeds. Therefore, we manually changed the first values of the code to 10 mm/s to be safe.

In Table 6.23 the rest of the printing settings are displayed. We will print with an speed factor of 30% to make sure that the material has time to cool down and the extrusion

	temperature (°C)
entry zone	210
compression zone	220
nozzle	175
bed	120

Table 6.22 : Temperature data of horizontal panel N°1

factor will remain in 100% in other words, we don't change the extrusion value that was given by the SilkWorm plug in of Grasshopper. Similarly to the simple cube, the fans will remain off for the printing of the base of the panel and then we will turn on the cooling system when it starts printing the single cells.

speed factor	30%
extrusion factor	100%
fans	off/on

Table 6.23 : Horizontal panel N°1 settings

The result of this first try can be seen in Figure 5.9. It is clear that it does not look like what we expected. It must be said that printing was aborted before finishing the code since several problems were encountered.

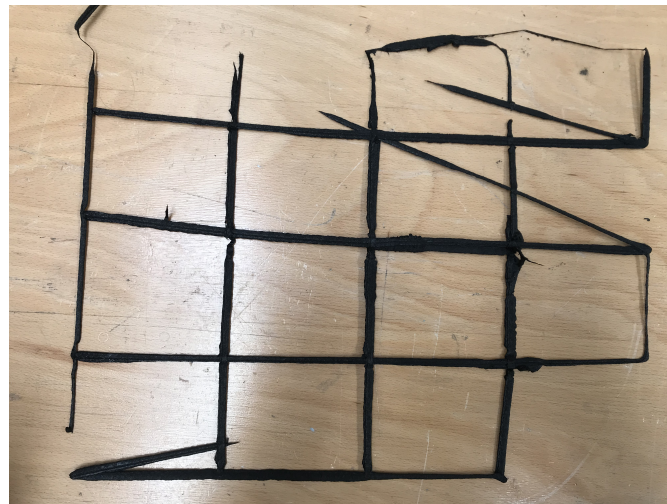


Figure 6.23 : Top view of the horizontal panel N°1

The code we wrote for this print was supposed to print the base of the horizontal panel, then start printing the single unit cells, and finally print the top, closing part of the panel. In Figure 5.9 it is seen how we only got to print the first part.

In the Grasshopper interface we stated the order of printing, how for the bottom base we wanted to print first vertical lines and then the horizontal ones to have the desired base, this was established before using the SilkWorm plug in. When printing the part this was accomplished, first the vertical lines and then the horizontal ones were done, the movement of the robotic arm was correct. However, the extruder did not work as expected. As it can be seen in Figure 5.9 some of the segments of the base lines did not print, the robotic arm went through the points but the nozzle did not extrude any material. In other parts of base the extruder went twice and the nozzle kept extruding material both times. Also when the robotic arm moved diagonally to get to another vertical line, the nozzle kept extruding material when in reality it shouldn't. This is the main reason why we aborted the print. We decided to stop printing and check the code and see if there were any mistakes.

It is important to remember how the code was written. First the horizontal lattice was designed in Grasshopper interface, and after deciding the order and the correct way of printing, the toolpath was inserted into the SilkWorm plug in to create the corresponding G-Code. The SilkWorm plug in is like a black box, we don't really know what happens inside, we insert a settings dictionary with some inputs about the layer height we want, first layer of extrusion width... and with this information the plug in makes the G-Code.

We open the G-Code of the horizontal panel and check the commands that the SilkWorm plug in generated, seen in Figure 6.24. It can be seen how the first vertical line has been correctly done, since there are five commands, one command for each point that makes the vertical line. However, for the printing of the second line we find twice the amount of commands, and all of them have extrusion values. I have highlighted with the same colour when the command repeats itself. This explains why when we were printing, some segments of the vertical lines were printed twice. However it does not explain why when we were moving from one segment to the other the nozzle kept extruding material. Since the SilkWorm plug in works like a black box I don't know why it rewrote some commands twice and others (like the first line) were kept intact.

Due to the short time available for printing, I could not look into the grasshopper file closely to see the mistake, therefore I deleted the repeated values manually, the result is seen in Figure 6.25, it is saved as "horizonta_panel_manually_modified.gcode".

Once all the printing was done I went through the Grasshopper file again to try and find the mistake. Looking at the toolpath I developed seen in Figure 5.14 in Section 5.3.2 I think that I found the mistake. This can be seen in more detail in Figure 6.26. This shows two different approaches to get the vertical and horizontal lines. At the top part of the Figure 6.26 it can be seen that the command "join curves" was used, this is the command that was originally used in Section 5.3.2, I have added a panel to see the output when using this command. It can be seen that this command creates five outputs, two lines (first and last output) and three polylines. The polylines are represented in the

```

G90 ; use absolute coordinates
G21 ; set units to millimeters
G92 E0 ; reset extrusion distance
M104 S280 ; set temperature
M109 S280 ; wait for temperature to be reached
G1 Z0.0 F360 E1
G92 E0
G1 F1200 X605.49 Y241.47 Z2
G1 F1200 X497.37 Y241.47 Z2 E119.8
G1 F1200 X389.25 Y241.47 Z2 E119.8
G1 F1200 X281.14 Y241.47 Z2 E119.8
G1 F1200 X173.02 Y241.47 Z2 E119.8
G92 E0
G1 F1200 X605.49 Y325.01 Z2
G1 F1200 X497.37 Y325.01 Z2 E119.8
G1 F1200 X389.25 Y325.01 Z2 E119.8
G1 F1200 X281.14 Y325.01 Z2 E119.8
G1 F1200 X173.02 Y325.01 Z2 E119.8
G1 F1200 X281.14 Y325.01 Z2 E119.8
G1 F1200 X389.25 Y325.01 Z2 E119.8
G1 F1200 X497.37 Y325.01 Z2 E119.8
G1 F1200 X605.49 Y325.01 Z2 E119.8
G92 E0
G1 F1200 X605.49 Y408.56 Z2
G1 F1200 X497.37 Y408.56 Z2 E119.8
G1 F1200 X389.25 Y408.56 Z2 E119.8
G1 F1200 X281.14 Y408.56 Z2 E119.8
G1 F1200 X173.02 Y408.56 Z2 E119.8
G1 F1200 X281.14 Y408.56 Z2 E119.8
G1 F1200 X389.25 Y408.56 Z2 E119.8
G1 F1200 X497.37 Y408.56 Z2 E119.8
G1 F1200 X605.49 Y408.56 Z2 E119.8
G92 E0
G1 F1200 X605.49 Y492.1 Z2
G1 F1200 X497.37 Y492.1 Z2 E119.8
G1 F1200 X389.25 Y492.1 Z2 E119.8
G1 F1200 X281.14 Y492.1 Z2 E119.8
G1 F1200 X173.02 Y492.1 Z2 E119.8
G1 F1200 X281.14 Y492.1 Z2 E119.8
G1 F1200 X389.25 Y492.1 Z2 E119.8
G1 F1200 X497.37 Y492.1 Z2 E119.8
G1 F1200 X605.49 Y492.1 Z2 E119.8
G92 E0
G1 F1200 X173.02 Y575.65 Z2

```

Figure 6.24 : G-Code horizontal panel

```

G90 ; use absolute coordinates
G21 ; set units to millimeters
G92 E0 ; reset extrusion distance
M104 S280 ; set temperature
M109 S280 ; wait for temperature to be reached
G1 X-200 Y200 Z2 F1200 E1
G92 E0
G1 F1200 X605.49 Y241.47 Z2
G1 F1200 X497.37 Y241.47 Z2 E119.8
G1 F1200 X389.25 Y241.47 Z2 E119.8
G1 F1200 X281.14 Y241.47 Z2 E119.8
G1 F1200 X173.02 Y241.47 Z2 E119.8
G92 E0
G1 F1200 X605.49 Y325.01 Z2
G1 F1200 X497.37 Y325.01 Z2 E119.8
G1 F1200 X389.25 Y325.01 Z2 E119.8
G1 F1200 X281.14 Y325.01 Z2 E119.8
G1 F1200 X173.02 Y325.01 Z2 E119.8
G92 E0
G1 F1200 X605.49 Y408.56 Z2
G1 F1200 X497.37 Y408.56 Z2 E119.8
G1 F1200 X389.25 Y408.56 Z2 E119.8
G1 F1200 X281.14 Y408.56 Z2 E119.8
G1 F1200 X173.02 Y408.56 Z2 E119.8
G92 E0
G1 F1200 X173.02 Y575.65 Z2
G1 F1200 X281.14 Y575.65 Z2 E119.8
G1 F1200 X389.25 Y575.65 Z2 E119.8
G1 F1200 X497.37 Y575.65 Z2 E119.8
G1 F1200 X605.49 Y575.65 Z2 E119.8
G92 E0
G1 F1200 X605.49 Y492.1 Z2
G1 F1200 X497.37 Y492.1 Z2 E119.8
G1 F1200 X389.25 Y492.1 Z2 E119.8
G1 F1200 X281.14 Y492.1 Z2 E119.8
G1 F1200 X173.02 Y492.1 Z2 E119.8
G92 E0
G1 F1200 X281.14 Y575.65 Z2

```

Figure 6.25 : modified G-code

G-Code as the repeated values, since it prints the line and goes back again extruding material, seen in the code of Figure 6.24.

If we look at the bottom part of the same Figure 6.26, we see that the command "join curves" has been deleted, and the corresponding panel shows that the output are five straight lines. We no longer have polylines which means that the robotic arm will go only once through each point hence the nozzle will only extrude once each segment. However, when simulating the code, I saw that it only printed some segments and not the whole line, therefore, the command "join curves" was necessary. In conclusion, I think that the problem must be inside the SilkWorm plug in, maybe it is not supposed to be used for this kind of free form printing.

Apart from this incident, the printing of the horizontal panel N°1 shown in Figure 6.23 had a few more mistakes. The bed adhesion was very bad, in some cases we had to use tape so that the print would not move, has it has been said before the main reason for this is the material chosen, PP is very prone to warping and it is difficult to adhere, increasing the bed temperature with respect to other prints shown in Section 6.1 did not seem to improve the adhesion. Also, the printing time was very long, maybe in the

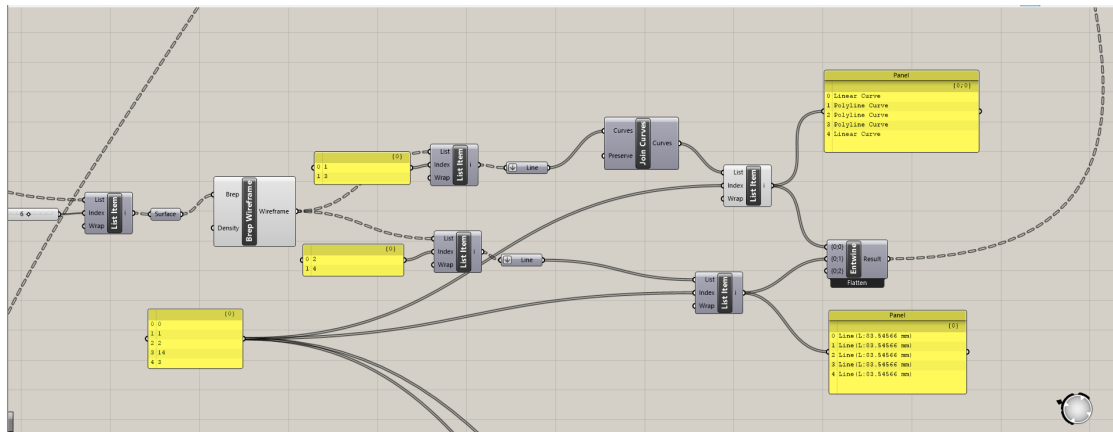


Figure 6.26 : Modified Grasshopper file

next try we could try to speed up a little bit the process, since we are already on a tight schedule and we have to print everything in three days.

Since we had to abort the rest of the printing we did not have a chance to check if the rest of the code was correctly implemented, this will be done in the following try.

6.2.2 Horizontal panel N°2

For this second print of the horizontal panel, the temperatures chosen are displayed in Table 6.24, the temperatures are very similar to the ones we used in Section 6.2 since we were happy with the consistency of the material being extruded. Regarding the bed temperature, we have not changed it. We know that we will encounter bed adhesion problems due to the material chosen and that varying the temperature of the bed a little bit will not make any difference.

	temperature (°C)
entry zone	195
compression zone	210
nozzle	175
bed	120

Table 6.24 : Temperature data of horizontal panel N°2

The other configurations options are shown in Table 6.25. It can be seen that we have increased the printing speed factor to 50% for the bottom printing to save time and for the printing of the unit lattice cells we will use 30%. A new parameter appears, the smooth command is off, this has been already mentioned in Section 6.1.11, but it was first used in this print and later we reprinted the simple cube using this configuration. By deleting the smooth command in the RobotDK code we stop wasting material, the material will only be extruded when it is said in the code. The smooth command is

used in normal 3D printing to join and cure the print. Here we are performing individual commands, we don't want the nozzle to extrude material when the robotic arm is moving away from a certain point. It is a shame that we did not take into account this option earlier, but it will be interesting to see how the structure is now printed.

speed factor	50% and 30%
extrusion factor	100%
fans	off/on
smooth	off

Table 6.25 : Horizontal panel N°2 settings

The result of the second printing can be seen in Figure 6.27. First of all, we can see how deleting the smooth command was a good idea, since the base of the horizontal panel is perfect. We no longer have extra filament lying around.

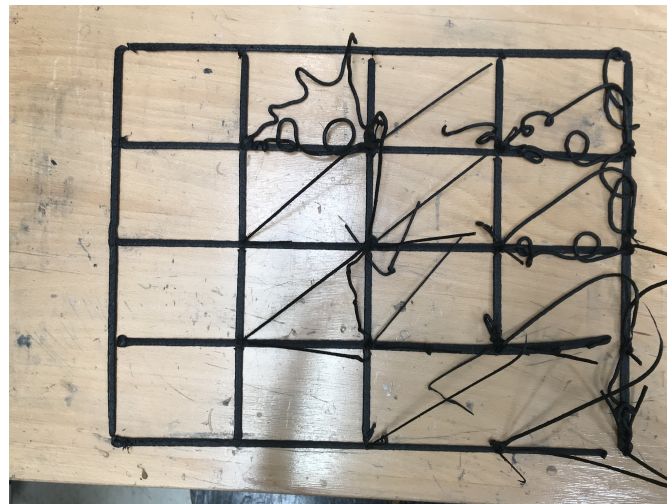


Figure 6.27 : Top view of the horizontal panel N°2

Regarding the bed adhesion, we still have problems sticking the print to the bed. The normal 3D printing slicers have an option to fight against bed adhesion problems, specially if the printing base of the product is very small and thin. A brim is printed, which circles many times around the base of the object that we want to print. Also, a raft can be printed, which is a thin platform printed underneath the desired object, which is later removed. For our free form printing we could also implement a brim or a raft to solve this adhesion problem.

The part being printed can be seen in Figure 6.28. it shows how the vertical struts are more or less staying in place. As it was said before, when printing the lattice cells the speed was decreased to 30%. However this was not slow enough for the material to cool

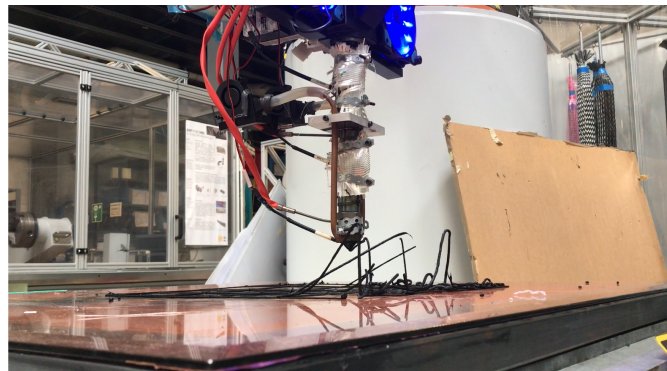


Figure 6.28 : Printing of the horizontal panel N°2

down, it is seen in Figure 6.28 that the right vertical struts, which were printed first have not solidify properly. Therefore, the speed was decreased to 15% and we finally get the right printing speed, it can be seen in a video that has been attached to the thesis called "horizontal_panel.mov".

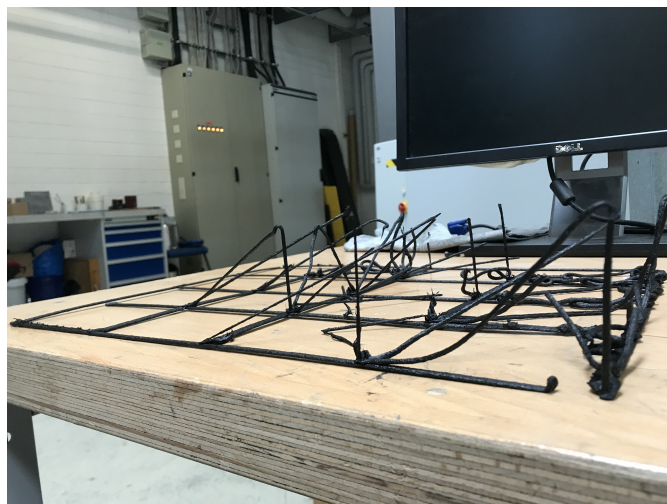


Figure 6.29 : Side view of the horizontal panel N°2

Another problem we encountered when printing the horizontal panel were the collisions. There was an error in the code implemented and it always tried to print twice the vertical struts, this is why in Figure 6.29 it can be seen how there are some vertical struts are broken and lying on the floor. Since this structure takes a lot of time to be printed and we are short on time, we will not print this structure again but we will try this configuration on the simple cube seen in sections 6.1.10 and 6.1.11. We had a slot of three days to print all the structures, then the pulsar extruder was disassembled and moved to another robotic arm.

Even though that this panel will not be printed again, I wanted to see where the code went wrong. The error is shown in Figure 6.30, I took the same list item that for the single cube code and I did not realise that when morphing the toolpath of the single cube into the horizontal panel it rotated the toolpath. Once I discovered this, I changed the order of

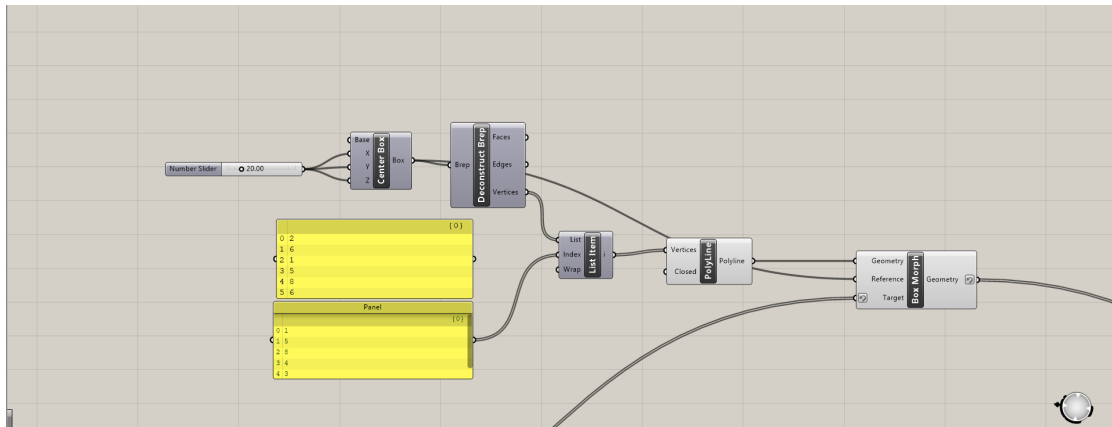


Figure 6.30 : Modified code in Grasshopper interface

vertices of the list item which is seen in Figure 6.30 to one that will 100% match the desired toolpath in the horizontal panel, the code is saved as "horizontal_modified.gcode". I would have liked to verify that with this code, the middle part of horizontal panel was correctly printed but it was not possible to use the printer.



Figure 6.31 : Unit cell horizontal panel

Figure 6.31 shows in more detail a single cell of the horizontal panel. It was the best unit cell that was printed. It can be seen that thanks to the reduction of speed to 15% the vertical and diagonal struts were perfectly printed. Also, the bonding to the bottom layer was great. Since the size of the single cells were not squares but rectangles, the diagonals were not as steep as with the single cube and there was less risk of collision.

Overall, I am happy with the results, even though that we did not finish printing the whole structure because it took too long. In the future it would be nice to try it again also with other filaments to see how the print behaves and if it is able to stick to the table. In reality, this structure is a collection of single lattice cells, so it makes sense that if we are short on time we focus on printing just one lattice cell with different settings instead

of printing a bigger structure. Once the configuration of the single cube is perfect the settings can be automatically used for this print.

6.3 Orientation change

The final code that I created was the pentagon structure. This structure is to be printed with the tilting and twisting of the KUKA arm, rotation values to the x and y axis of the robotic arm have been calculated. Since the whole robotic arm will rotate and bend we have to very careful when printing this structure because this has never been done in this robot before. Also the speed will be controlled manually at all moments by my supervisor so in case anything goes wrong it will be immediately stopped. Several tries have been done and they are explained in detail in the following sections.

The workflow followed for this last try is the same that for the previous prints. The code generated is loaded into the RobotDK interface, we check that there are no collisions and that everything works fine and we are ready to print. It was mentioned in Section 5.3.3 that the B values of the code had to be rotated -90° because the tool head of the Grasshopper interface an the tool head of the actual robotic arm were not aligned.

6.3.1 Pentagon N°1

The temperature settings chosen for this first print can be seen in Table 6.26. They are the same ones that we used for the horizontal panel .

	temperature (°C)
entry zone	195
compression zone	210
nozzle	175
bed	120

Table 6.26 : Temperature data of pentagon N°1

It has been said before that the speed of the robotic arm will be given manually and carefully controlled at all times. Since the KUKA robot is able to move up to 2 m/s we have to be extremely careful and monitor all the movements. If we see that something does not behave as it should we will automatically abort the printing. The rest of the printing settings can be seen in Table 6.27. Once again, we will only use the fans when printing the upwards movement of the pentagon, the base will be done with the fans off to favour base adhesion. Since we are going to print at very low speeds, the extrusion factor has been increased to 200%, double the filament will be extruded.

speed factor	manual
extrusion factor	200%
fans	off/on
smooth	on

Table 6.27 : Pentagon N°1 settings

The code we created works well since it moves as we expected. It is very scary to see how the robotic arm bends. We had to control at all times that it did not collide with anything. There is too much filament coming out of the nozzle and it does not have enough time to solidify, this is why the diagonal is not forming. Maybe, the extrusion factor used was too big.



Figure 6.32 : Front view of pentagon N°1

The result of this first try can be seen in Figures 6.32 and 6.33. First of all, not all the structure was printed. We only attempted to print the first two diagonals. When we arrived to the bottom of the second diagonal we had to abort the printing because the tip of the extruder was in contact with the printing table, a lot of pressure was on the table which could end up breaking. Before going any further we decided to stop the printing. To prevent the collision between the printing table and the nozzle we can give an offset to the bottom planes of the diagonals, so the nozzle has enough room to bend and tilt, this will be implemented in the next try.

In the front view of the print, seen in Figure 6.32 it is easy to see how the print did not adhere correctly to the bed, in this case it was not possible to use tape to fix it to the table since we had to be very careful and at a safe distance from the robotic arm movements. The top view seen in Figure 6.33 shows how the bottom pentagon was correctly printed, the amount of filament used was okay, so in future prints, for the bottom part we will use



Figure 6.33 : Top view of pentagon N°1

200%. Also, we have to improve the extrusion of the diagonal parts, by decreasing the amount being extruded it might be able to solidify faster and in the correct way. It would be great if we could check that all the structure can be printed without any problem, that all rotation of the axis does not interfere with anything.

6.3.2 Pentagon N°2

The temperature settings used for the printing of pentagon N°2 are seen in Table 6.28, they are the same ones used for the previous print since we were happy with the characteristics of the prints, we have already checked that changing the bed temperature does not help with bed adhesion therefore, we will keep the same one as before and discuss other ways to improve the bonding with the printing bed in the following chapter of future improvements.

	temperature (°C)
entry zone	195
compression zone	210
nozzle	175
bed	120

Table 6.28 : Temperature data of pentagon N°2

Regarding the other printing settings shown in Table 6.29, it can be seen that there are two extrusion factors: 200% will be used for the base, because using less extrusion factor might lead to worse bed adhesion. 100% extrusion factor will be used for the rest of the print. In the previous print, Section 6.3.1, 200% extrusion factor was used for the

up and down parts and we saw that there was too much filament, decreasing this factor will hopefully lead to a better result.

speed factor	manual
extrusion factor	200% and 100%
fans	off/on
smooth	on

Table 6.29 : Pentagon N°2 settings

The printing of Pentagon N°2 can be seen in Figure 6.34, it has been extracted from a time lapse video available with the thesis saved as "pentagon2.mov". It shows how the robotic arm is tilting to print the desired part. Also, it is seen the effect of having the smooth command. If we look closely, it can be seen that when the robotic arm is moving towards the table, the nozzle is extruding filament.

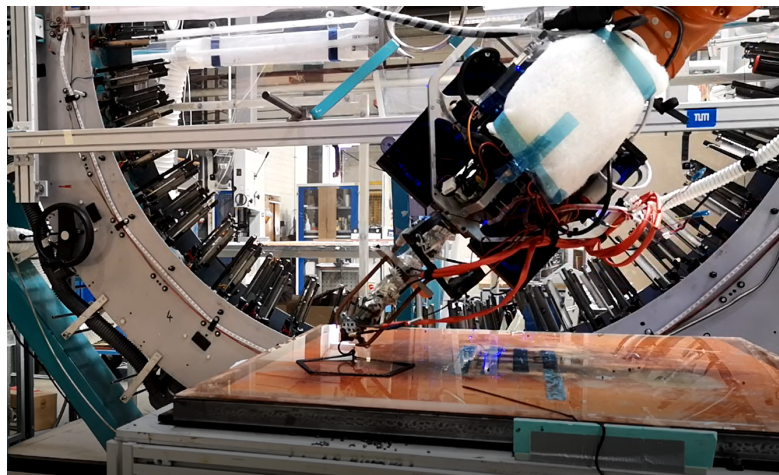


Figure 6.34 : Printing of pentagon N°2

Figures 6.35 and 6.36 show the final result of this print. In the previous print we had a collision problem between the printing table and the nozzle of the extruder, the nozzle did not have enough room to bend and kept printing. To solve this problem we have applied an offset of 2 mm to the bottom planes of the pentagon. However, this did not work as well as we expected, the printing nozzle kept contacting the table with quite a lot of pressure and to avoid the breakage of the table we aborted the print.

Once again, we were not able to check if the robot could print the whole structure without colliding with other parts of the system. It is better to be extra careful than later regret it if we break the table or if the robot becomes uncontrolled.

Changing the extrusion factor for the up and down part was a good idea, Figure 6.35 shows how the amount of filament being extruded is adequate, even though that the



Figure 6.35 : Front view of pentagon N°2

material is not cooling down in the expected way. The robotic arm is following the desired toolpath, and at the same time the nozzle is extruding the material. However, the material is not cooling down fast enough and the gravity pulls it down and ends up looking looking very wobbly. To solve this, maybe we should print at lower speeds, change the cooling system or find a more suitable material, with a higher glass transition temperature that solidifies right after coming out of the nozzle.



Figure 6.36 : Top view of pentagon N°2

6.3.3 Pentagon N°3

We decided to make one more try of the horizontal panel, keeping the same configuration that in Section 6.3.2, shown in Tables 6.28 and 6.29. The speed factor of the robotic arm will be controlled manually. As it was said in the previous section, for the up and

down struts of the pentagon we will reduce even more the velocity, allowing the material to cool down properly. The fans will be off for the printing of the base, to favour the bed adhesion and on for the rest of the printing, to favour the cooling down of the filament.

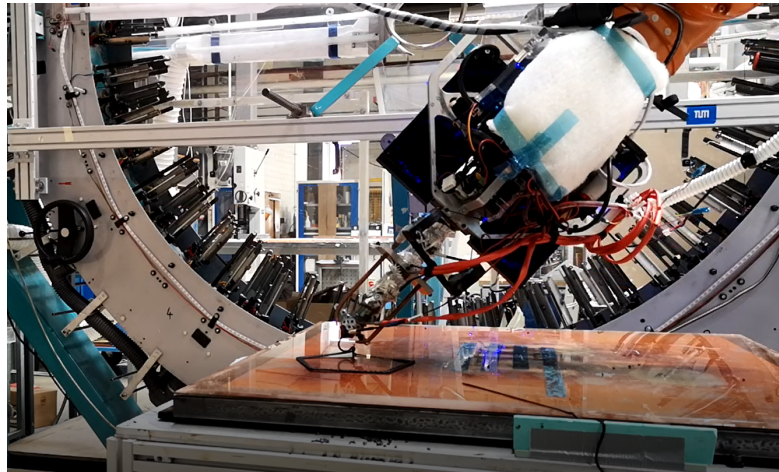


Figure 6.37 : Printing of pentagon N°3

Figure 6.37 shows a shot of the printing video of pentagon N°3 which is available and saved as "pentagon3.mov". It is seen how the KUKA arm tilts and extrudes material. For this print we used the same code modification as the one in Section 6.3.2, this code has an offset of 3mm in the bottom planes of the diagonal struts. In other words, instead of going down to Z2 as originally planned, it goes to Z5 so the nozzle of the pulsar extruder has enough room to bend and rotate to the next plane without colliding with the printing bed. However, it did not work as we expected, since once again it kept colliding into the printing table and we had to abort the print for safety reasons. We were not able to see the whole structure being printed and check if all the sections were able to print without colliding with anything else. .



Figure 6.38 : Front view of pentagon N°2

For the future, we could try the code in a bigger printer, that is fully isolated and cannot collide with anything around, so we don't have to pay as much attention to the environment and we can focus more on the printing

The results of this last try are seen in Figure 6.38 and 6.39. First of all, the adhesion of the print to the heated bed was bad, and we could not do anything when it was printing to fix it. In other sections we have used tape to stick the print to the table but it was dangerous to be near the table when the robotic arm was tilting and rotating.



Figure 6.39 : Top view of pentagon N°2

In general, the results of this last try were similar to the ones in Section 6.3.2. The material is not cooling fast enough when it does the diagonal struts. Moreover, the robotic arm does follow the correct toolpath. But, if we look closely to the tip of the extruder, it seems like there is an offset between the tool center point and the actual center point of the pulsar extruder and this might cause the material to be extruded in a position different to the one that it was supposed to be. In the future, we could try to print this structure in a different machine, to check if this problem keeps appearing or not.

7 Future improvements

If this investigation is retaken in the future, there are some aspects that could be improved to get better results. Most of them have been mentioned throughout the validation in Section 6.

7.1 Cooling system

One of the main goals of the thesis was to design a cooling system that would allow rapid cooling of the material being extruded. Several designs were proposed but finally we chose a system composed of two radial fans that blew air into two pipes next to the nozzle of the extruder.

The assemble of the cooling system is explained in more detail in Section 3. It is mentioned how we had to change the orientation of the silicone tubes that connected the copper pipes and the fans because there was not enough air coming out of the pipes. Changing the orientation of the system meant that the part created in CATIA and 3D printed was no longer adequate, it could not be screwed into the the robotic arm. We had to use some tape to make sure that it did not move.

For the future, before 3D printing the parts, fluid simulations could be made to ensure the correct flow of air through the pipes. The orientation, length and size of the silicone tubes has a big impact on the way the flow is distributed, therefore, fluid simulations would be very helpful. Also, it would be interesting to see if more powerful radial fans can be purchased, to get more air at the bottom of the pulsar extruder. Since the new fans will be more powerful, they will probably be bigger, and so the support structures designed in CATIA will change. This is not a problem since the CATIA design is fully parametric.

During the printing of the simple cubes we had some collision problems that we were not expecting. The cooling system, more specifically the copper pipes collided with the material being extruded, breaking it. In order to solve this, the pipes could be placed somewhere else, further away from the nozzle, so they do not interfere with the filament. However, moving the pipes away could decrease the amount of air that the material extruded receives to cool down. Changing the size of the pipes, the diameter could be a good option too. Making them smaller will reduce the chance of colliding and also having a smaller outlet diameter will increase the outlet velocity of the air which could be beneficial.

7.2 Bed adhesion

Bed adhesion has been one of the main concerns throughout the whole validation of the project. In many prints we have seen how the material did not stick to the table and as a result the whole print moved. In some cases, we used tape to stick the base of the print to the table and make sure that it did not move. Ideally, the material should be able to stick to the table without us having to constantly check and add tape. There are several options that we could not implement but could be used in the future:

- **Printed brim:** Is a special type of skirt, it will be attached to the edges of the desired print, similar to the brim of a hat. This will help to push the edges of our part down and prevent the warping and bed adhesion problems that we might have. The brim increases the surface area of the part, more surface area contacting the bed, more force pushing down, hence better results.
- **Printed raft:** A raft is a whole bottom layer printed underneath the desired part. It removes the problematic bottom layer problem. Since we are printing single struts as base for our lattice structures this might not be the best option, since it will be difficult to remove the raft from the lattice cell when the printing is done.
- **Adjust fan speed:** This is the only option we have implemented throughout the printing. When printing the base layer of our designs we turned off the fans, so the filament had more time to adhere to the table. In some prints we saw the difference between having the fans on and off for the first layer and the differences were noticeable, but not good enough.
- **Coating the bed:** Adding a coat to the printed bed will act as glue and stick the printed part to the table.

7.3 Material

Another big concern when validating the project was the material used. It was discussed in Section 4 that the material used would be PP with glass fibers. This choice was mainly done due to the availability of the materials. PP is a semi-crystalline thermoplastic which is difficult to thermoform and bond. It has been seen in the validation, in Section 6 how when printing the top struts of the lattice structures they did not bond well with the already printed vertical and diagonal struts. Also it is very prone to warping and tends to shrink which has been seen in the different prints done.

Moreover, when printing the final parts, the pentagon structures, the material did not solidify as fast as we wanted and this meant that the whole structure broke down. This can be solved with a better cooling system which has already been discussed, but

changing the material would also help. Having a material with a high glass transition temperature means that when coming out of the nozzle and being in contact with the temperature of the environment it solidifies immediately. Amorphous thermoplastics such as ABS with carbon fiber reinforcement or PC could be a good alternative. It would be interesting to see how the results vary with a different choice of material.

7.4 Grasshopper

Using Grasshopper and Rhino for the generation of the code of the lattice structures has been very interesting and I have learnt a lot since I did not know how to use this programming language. I took my some time to get used to the interface and learn how to use the commands, but once this initial problem was solved a large range of possibilities appeared.

Every day new plug in are created for Grasshopper that help you program and accomplish things that you never thought were possible. Overall, I think that the code I created for these prints was good, most of it was fully parametric. But in the future I would like to explore other options. The Silkworm plug in worked well with the simple cube since it was printing a continuous polyline. But, when we tried printing the horizontal lattice structure we saw that the code repeated some commands twice, and after revising the code we concluded that it was something inside the SilkWorm plug in. If this research is retaken in the future, I would like to look into the Silkworm plug in in more detail, or even find a more suitable plug in.

For the final code generation another plug in was used: KUKAlprc, whose output was the KUKA code. This code had to be manually modified to convert it into a G-Code. For the future, it would be great the develop a way to obtain the G-Code automatically in the Grasshopper interface.

8 Conclusions

Free form printing uses state of the art technology to enable the manufacturing of large scale 3D printing. In this thesis I have done an extensive research and in depth analysis of existing free form printing testing such as Branch Technology and AiBuild. I have analysed what they do and how they do it to try and establish a similar workflow that could print large lattice structures.

Thanks to this project I have learnt to program in Grasshopper, to control the movement of a robotic arm which I have found very interesting and I would like to learn more about it in the future. Besides computer programming, I have had the chance to bring up concepts from CAD and materials among others to obtain the whole printing system.

Throughout the thesis I have found some problems and difficulties. The most evident limitation was the lack of time for the validation part. We had only a few days to print and validate the code created, which meant that the problems that we encountered while printing had to be solved on the spot. As a result, some of the solutions that were proposed are not ideal and could have been much better. Nevertheless, the inconveniences are far from outweighing the priceless benefits that have come from the fulfillment of this research.

I believe that there is still much room for refinement and thanks to the improvements discussed in the previous section the results obtained will be much better. I would love to keep working on this topic on the future and see if the proposed improvements work.

In the future, I would focus more on the last part of the thesis, working with the rotation of the robotic arm which was truly fascinating to see in action. Also, the selection of materials has been very interesting. During my bachelor, I have had some subjects related to material science, but it has been during the development of this thesis that I have learnt the importance of selecting the adequate material. The results obtained would be quite different if we had chosen a different material.

The field of research is crucial for the development and growth of technology. Sometimes the validation of the research does not reflect desired results, but that does not mean that the research done is not useful. It is important to learn from the mistakes, make room for improvement and overall, keep on trying and working. I believe that 3D printing lattice structures will be a key feature on the non so distant future of engineering.

Bibliography

- [1] AJINJERU, CHRISTINE, VIDYA KISHORE, PENG LIU, JOHN LINDAHL, AHMED ARABI HASSEN, VLASTIMIL KUNC, BRIAN POST, LONNIE LOVE and CHAD DUTY: *Determination of melt processing conditions for high performance amorphous thermoplastics for large format additive manufacturing*. Additive Manufacturing, 21:125 – 132, 2018.
- [2] BOYD, R.PLATT: *Branch Technology*, 2015.
- [3] BRAUMANN, JOHANNES and SIGRID BRELL-ÇOKCAN: *Parametric Robot Control: Integrated CAD/CAM for Architectural Design*. In [ACADIA 2011 : integration through computation : proceedings of the 31st annual conference of the Association for Computer Aided Design in Architecture (ACADIA)], pages 242–251. 31st Annual Conference of the Association for Computer Aided Design in Architecture, Banff, Alberta (Canada), 13 Oct 2011 - 16 Oct 2011, Association for Computer Aided Design in Architecture, Oct 2011.
- [4] CAMPO, E. ALFREDO: *3 - Thermal Properties of Polymeric Materials*. In CAMPO, E. ALFREDO (editor): *Selection of Polymeric Materials*, Plastics Design Library, pages 103 – 140. William Andrew Publishing, Norwich, NY, 2008.
- [5] D. CAM, M.DESYLLAS: *AiBuild*, 2015.
- [6] ISHAK, ISMAYUZRI and PIERRE LAROCHELLE: *MotoMaker: a robot FDM platform for multi-plane and 3D lattice structure printing*. Mechanics Based Design of Structures and Machines, pages 1–18, 05 2019.
- [7] JIN, MINDE, REINER GIESA, CHRISTIAN NEUBER and HANS-WERNER SCHMIDT: *Filament Materials Screening for FDM 3D Printing by Means of Injection-Molded Short Rods*. Macromolecular Materials and Engineering, 303(12):1800507, 2018.
- [8] LIU, SHUTING, YINGGUANG LI and NANYA LI: *A novel free-hanging 3D printing method for continuous carbon fiber reinforced thermoplastic lattice truss core structures*. Materials Design, 137:235 – 244, 2018.
- [9] MACONACHIE, TOBIAS, MARTIN LEARY, BILL LOZANOVSKI, XUEZHE ZHANG, MA QIAN, OMAR FARUQUE and MILAN BRANDT: *SLM lattice structures: Properties, performance, applications and challenges*. Materials Design, 183:108137, 2019.

-
- [10] MANUFACTURING, STRATASYS DIRECT: *3D printing materials: choosing the right material fro your application*. Technical Report, 2011.
- [11] OXMAN, NERI, JARED LAUCKS, MARKUS KAYSER, ERIC T TSAI and MICHAL FIRSTENBERG: *Freeform 3D printing: Towards a sustainable approach to additive manufacturing*. 2013.
- [12] OZDEMIR, ZUHAL, EVERTH HERNÁNDEZ-NAVA, ANDREW TYAS, JAMES WARREN, STEPHEN FAY, RUSSELL GOODALL, I. TODD and HARM ASKES: *Energy absorption in lattice structures in dynamics: Experiments*. International Journal of Impact Engineering, 89, 11 2015.
- [13] PARK, SANG-IN, DAVID W. ROSEN, SEUNG KYUM CHOI and CHAD E. DUTY: *Effective mechanical properties of lattice material fabricated by material extrusion additive manufacturing*. Additive Manufacturing, 1-4:12 – 23, 2014. Inaugural Issue.
- [14] SHEN, HONGYAO, LINGNAN PAN and JUN QIAN: *Research on large-scale additive manufacturing based on multi-robot collaboration technology*. Additive Manufacturing, 30:100906, 2019.
- [15] SPOERK, MARTIN, CLEMENS HOLZER and JOAMIN GONZALEZ-GUTIERREZ: *Material extrusion-based additive manufacturing of polypropylene: A review on how to improve dimensional inaccuracy and warpage*. Journal of Applied Polymer Science, 137(12):48545, 2020.
- [16] YAZDI, MILAD, SEYYED ALI LATIFI ROSTAMI and AMIN KOLAHDOOZ: *Optimization of geometrical parameters in a specific composite lattice structure using neural networks and ABC algorithm*. Journal of Mechanical Science and Technology, 30:1763–1771, 04 2016.
- [17] ZHOU, J, P SHROTRIYA and W.O SOBOYEJO: *On the deformation of aluminum lattice block structures: from struts to structures*. Mechanics of Materials, 36(8):723 – 737, 2004. Mechanics of Cellular and Porous Materials.

List of Figures

2.1	Multi-robot 3D printing system [14]	5
2.2	ABS Filament freeform extrusion [11].	6
2.3	Branch Technology [2]	6
2.4	AiBuild [5]	6
2.5	MotoMaker [6]	7
2.6	Free hanging 3D printing [8]	8
3.1	Pulsar Extruder.	10
3.2	Design of the top structure	11
3.3	Design of the bottom structure	11
3.4	Side view of the top structure	12
3.5	Rear view of the top structure	12
3.6	Front view of the system	13
3.7	Closer shot at the bottom of the extruder	13
3.8	Final orientation of the top part	14
3.9	Final position of the structure	15
5.1	Integral wing [8]	20
5.2	Cellular Fabrication. Image courtesy of Branch Technology	21
5.3	Grasshopper workflow of simple cube	22
5.4	Simple cube in the Rhino interface	23
5.5	Settings dictionary of the extruder	23
5.6	Grasshopper workflow of simple cube	24
5.7	Grasshopper workflow of simple cube	24
5.8	Grasshopper workflow of simple cube	25
5.9	General workflow of the horizontal panel	25
5.10	Horizontal panel in Rhino.	26
5.11	Division of the horizontal panel.	26
5.12	Deconstructed horizontal panel into small squares	27
5.13	Twisted box	27
5.14	Grasshopper code for the bottom part of the panel	28
5.15	Bottom part toolpath seen in green	28
5.16	Simple cube in the Rhino interface	29
5.17	Middle section of the horizontal panel	29
5.18	vertical struts in the grasshopper interface	30
5.19	Vertical struts of the middle section	30
5.20	G-Code generation of the horizontal panel	31

5.21	Horizontal panel final toolpath	31
5.22	Horizontal panel in RobotDK	32
5.23	Plane orientation geometry in Rhino interface	33
5.24	Plane orientation workflow in Grasshopper interface	34
5.25	Plane orientation workflow in Grasshopper interface	34
5.26	Filtering of the necessary planes in Grasshopper	35
5.27	Planes of the pentagon in the Rhino interface	36
5.28	KUKA prc in Grasshopper interface	37
5.29	KUKA prc in the Rhino interface	37
5.30	KUKA prc in the Rhino interface closer look	38
5.31	KUKA prc CORE	38
5.32	Analysis of the movement of the KUKA robot	39
5.33	pentagon.src	40
5.34	Extrusion value	41
5.35	G-Code pentagon	42
6.1	Cube N°1	45
6.2	Cube N°1 top view	45
6.3	Printing of Cube N°2	47
6.4	Cube N°2	47
6.5	Printing of Cube N°3	49
6.6	Cube N°3	49
6.7	Printing of Cube N°4	50
6.8	Cube N°4	51
6.9	Printing of Cube N°5	52
6.10	Cube N°5	53
6.11	Cube N°6 front view	54
6.12	Cube N°6 side view	55
6.13	Front view of Cube N°7	56
6.14	Side view of Cube N°7	57
6.15	Front view of Cube N°8	58
6.16	Side view of Cube N°8	59
6.17	Front view of Cube N°9	60
6.18	Side view of Cube N°9	61
6.19	Front view of Cube N°10	62
6.20	Side view of Cube N°10	63
6.21	Front view of Cube N°11	64
6.22	Side view of Cube N°11	65
6.23	Top view of the horizontal panel N°1	66
6.24	G-Code horizontal panel	68

6.25	modified G-code	68
6.26	Modified Grasshopper file	69
6.27	Top view of the horizontal panel N°2	70
6.28	Printing of the horizontal panel N°2	71
6.29	Side view of the horizontal panel N°2	71
6.30	Modified code in Grasshopper interface	72
6.31	Unit cell horizontal panel	72
6.32	Front view of pentagon N°1	74
6.33	Top view of pentagon N°1	75
6.34	Printing of pentagon N°2	76
6.35	Front view of pentagon N°2	77
6.36	Top view of pentagon N°2	77
6.37	Printing of pentagon N°3	78
6.38	Front view of pentagon N°2	78
6.39	Top view of pentagon N°2	79

List of Tables

3.1	Data of the Radial Fan	10
6.1	Temperature data of Cube N°1	44
6.2	Temperature data of Cube N°2	46
6.3	Cube N°2 settings	46
6.4	Temperature data of Cube N°3	48
6.5	Cube N°3 settings	48
6.6	Temperature data of Cube N°4	50
6.7	Cube N°4 settings	50
6.8	Temperature data of Cube N°5	52
6.9	Cube N°5 settings	52
6.10	Temperature data of Cube N°6	53
6.11	Cube N°6 settings	54
6.12	Temperature data of Cube N°7	55
6.13	Cube N°7 settings	56
6.14	Temperature data of Cube N°8	57
6.15	Cube N°8 settings	58
6.16	Temperature data of Cube N°9	59
6.17	Cube N°9 settings	60
6.18	Temperature data of Cube N°10	61
6.19	Cube N°10 settings	62
6.20	Temperature data of Cube N°11	63
6.21	Cube N°11 settings	64
6.22	Temperature data of horizontal panel N°1	66
6.23	Horizontal panel N°1 settings	66
6.24	Temperature data of horizontal panel N°2	69
6.25	Horizontal panel N°2 settings	70
6.26	Temperature data of pentagon N°1	73
6.27	Pentagon N°1 settings	74
6.28	Temperature data of pentagon N°2	75
6.29	Pentagon N°2 settings	76

Annex

The CD contains the following folders:

- PDF and LATEX files of the thesis with a sub-folder with all the images of the work.
- CAD Models of the cooling system.
- Grasshopper files.
- Rhino files.
- Codes generated for the validation.
- Videos of the validation process.
- PowerPoint presentation of the project.
- Citavi files with all the bibliography.