Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences

A safe and efficient path planning framework for conformal fused filament fabrication using a manipulator arm

A dissertation presented by

Ma. Consuelo Rodríguez Padilla

Submitted to the
School of Engineering and Sciences
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Engineering Science

Major in Additive Manufacturing and Robotics

Monterrey Nuevo León, December 2, 2022

Dedication

Firstly, thanks to my parents, my husband Salvador, my kids Salvador and Renata, for their support and help to achieve my professional goals. Thanks to my older brother Daniel, who was always an inspiration for me and who unfortunately passed away on June 3rd, 2022, during the last semester of my studies.

Acknowledgements

My studies would not have been possible without the scholarship granted by the Tecnologico de Monterrey on tuition and the Consejo Nacional de Ciencia y Tecnologia (CONACyT) for living. I really appreciate it and I am infinitely grateful. Thanks to the graduate program directors of the Tecnologico de Monterrey for leading one of the best universities in Mexico.

Thanks Dr. Neale and all the involved academic personnel for accepting me to the DCI program, their kindness, support, and readiness were always so distinctive.

Thanks to the personnel involved in the scholarship process, Angelica Garza Uranga and Bertha Sofía Villareal Gonzalez, for all the support during my studies and for making a complicated paperwork into a painless procedure.

Special thanks to my advisor Dr. Armando Román Flores for their unconditional support and patience. Even the difficulties during my studies, Dr. Armando never stopped believing in me and that kept me motivated. Thanks to Dr. Jose Luis Gordillo for agreeing to be my advisor and for sharing his knowledge with me. I learned so much!

Thanks to the significant professors that contributed to enrich my knowledge and made me a better person: Dr. Dr. Armando Roman, Dr. Jose Luis Gordillo Moscoso, Dr. Carlos Vázquez, Dr. Enrique Cuan, Dr. Leopoldo Cárdenas Barron, Dr. Gerardo Castañón, Dr. César Vargas, Dr. Federico Guedea, and Dr. Hugo González Hernández.

A safe and efficient path planning framework for conformal fused filament fabrication using a manipulator arm

By

Ma. Consuelo Rodríguez Padilla

Abstract

As opposed to flat or planar extrusion additive manufacturing, the benefits of multi-plane and curved fused deposition of material are conclusive; however, several issues need to be considered and solved when a robotic manipulator is used for the deposition of material. The path and motion planning for printing using robotics need considerations to guarantee adequate results. This work presents the projection of a printing trajectory on a tessellated surface and a Reinforcement Learning strategy that optimizes the angular displacement of joints. The validation of the strategy is presented under simulated conditions inserting different obstacles for a projected zigzag printing pattern on a curved surface. Results show that this approach can choose the optimal inverse kinematic solution to optimize the movement of the main joints of a robot with six degrees of freedom while avoiding different obstacles. The strategy was tested on several actual printings of complex patterns on different curved surfaces using a manipulator arm UR3. Even thought the applicability of lattice manufacturing suggested here, the framework developed and software implemented and validated may be used for any application where a very precise conformal trajectory needs to be followed using a manipulator arm or any multi-axis system saving programming time.

List of Figures

Fig. 2-1 Iso-planar (Cartesian) tool path generation. 19

Fig. 2-2 Points at the intersection between edges of triangles and orthogonal planes. . 20

Fig. 2-3 Generation of curved slicing layers by offsetting the z-value of vertices of the upper surface. 23

Fig. 2-4 Projection of zig-zag trajectory for surface finishing (smooth): vertices of triangles of tessellated target surface are projected to a spacing grid XY and the grid points, which belong to a certain triangle, are projected back to the target surface. 23

Fig. 2-5 A tessellated object_1 is conformally attached to a second tessellated object_2 24

Fig. 3-1 A solid model is created by revolving a profile curve around an axis. 27

Fig. 3-2 Some infill patterns found in Cura V4.5 slicing software. 28

Fig. 3-3 Points $P1 - P4$ that define the sinusoidal waveform in X-axis and Y-axis. 29

Fig. 3-4 User-defined parameters for the complete sinusoidal pattern for lattice. 30

Fig. 3-5 Layers of the printing sinusoidal trajectory and the printing direction. 30

Fig. 3-6 Sinusoidal waveforms are translated to a cylindrical parametric surface defined using three points $P1 - P3$. 31

Fig. 3-7 a) Parametric profile of the support, b) sinusoidal waveform running along the X-axis, and c) sinusoidal translated to the parametric profile of the cylindrical section. 31

Fig. 3-8 View X-Y of sinusoidal lattice for printing (left), and tridimensional view of lattice translated to the parametric profile (right)... 32

Fig. 3-9 Steps followed for the experimental validation of sinusoidal pattern on an analytical surface using a conventional Ender-3 printer. 32

Fig. 3-10 Surface from three sketches profiles.. 33

Fig. 3-11 Steps for printing a lattice using a zigzag trajectory on an analytical surface. 34

Fig. 3-12 Actual printed lattice on an analytical surface. 34

Fig. 3-13 Different trajectories generated on analytical surfaces using a conventional Ender-3 printer. 34

Fig. 3-14 Planar geometry with bending characteristics. 35

Fig. 3-15 Design of a bendable planar geometry. 35

Fig. 3-16 Two layers of conductive ink were applied and cured for 24hrs. 36

Fig. 3-17 Conductive ink was detached after bending the printed object 36

Fig. 3-18 Channels were added to the original geometry design for embedding a conductive wire. 37

Fig. 3-19 Detail of the wire embedded into the channels and the depth of base and channels. 38

Fig. 3-20 The piece is finished with wire embedded. 38

Fig. 3-21 The frequency response of the device was measured using a NI MyDAq board. 39

Fig. 3-22 When no folding took place, the phase was nearly -79 degrees at 10kHz. 39

Fig. 3-23 When partial folding took place, the phase was nearly -74 degrees at 10kHz. 40

Fig. 3-24 When significant folding took place, the phase was nearly -70 degrees at 10kHz. 40

Fig. 3-25 Magnetic fields generated when passing a current through a wire................ 41

Fig. 3-26 Multilayers of coils to produce a magnetic field. 41

Fig. 3-27 Simulation of the magnetic field generated in the inductor with five layers alternated with conductive and non-conductive material. 42

Fig. 3-28 Printing trajectories for the non-conductive and conductive layers of the inductor. 42

Fig. 3-29 Procedure for the manufacturing of the inductor, alternating non-conductive and conductive material. 43

Fig. 3-30 Observing the presence of the magnetic field using a compass. 44

Fig. 3-31 Channels incorporated into the design of pressure sensor for embedding conductive thread. 44

Fig. 3-32 Printing process paused to embed the conductive thread. 45

Fig. 4-1 Points in order are projected onto a curved tessellated surface. 47

Fig. 4-2 Flow diagram of the algorithm, where a point crosses a triangle. 47

Fig. 4-3 Experimental results of a lattice projected on an arbitrary surface. 48

Fig. 4-4 Commands inside the Gerber copper file (left) that produce the conductive coil track do not follow a continuous trajectory (right). 49

Fig. 4-5 PCB generated using upverter.com 49

Fig. 4-6 At the left, is the data contained in the copper Gerber file. On the right, it is the meaning of the two first commands. 50
Fig. 4-7 Suggested algorithm which divides the PCB into 3×3 sections to speed up the searching loop to find an optimal path printing. 51
Fig. 4-8 Mathematica software used to generate the geometry corresponding to a saddle surface according to $(4-1)$. 52
Fig. 4-9 Conductive tracks extracted from the Gerber file and projected on tessellated saddle geometry. 52
Fig. 4-10 Simulation of the trajectories of conductive tracks extracted from a Gerber copper file and then projected on a saddle surface. 53
Fig. 4-11 View in detail where each projected point has its surface normal vector. 53
Fig. 4-12 First and second iterations for a) Hilbert pattern and b) re-entrant pattern using L systems. 54
Fig. 5-1 System implemented for robotic curved conformal 3D printing. 56
Fig. 5-2 Extruder attached as a tool to the manipulator's arm 57
Fig. 5-3 Signals to the Arduino-based board to control the task of printing. 58
Fig. 5-4 Connection between the thermistors (located at the extruder and the build platform) and the RAMPS board. 59
Fig. 5-5 Characterization of the thermistor using an infrared thermometer as reference 60
Fig. 5-6 Electrical configurations to control the heater 60
Fig. 5-7 Motor driver for NEMA-17 for the extrusion of filament. 61
Fig. 5-8 Dimensions and configuration of arm robot UR3/CB3 64
Fig. 5-9 Configuration of the extruder as TCP of the robot 65
Fig. 5-10 Configuration of the build platform as a new work object to having a new origin with respect to the {Base} of the robot 67
Fig. 5-11 Parameters acquired during the build platform configuration. 68
Fig. 5-12 Reference frames for DH analysis of robot UR3/CB3 70
Fig. 5-13 The nomenclature is used according to the D-H convention to determine the joints' frames. 70
Fig. 5-14 Flow diagram of the execution of the developed and implemented software. 71

Fig. 5-15 Flow diagram of path and motion planning for robot UR3 72
Fig. 5-16 Up to eight solutions for the inverse kinematics of UR3 73
Fig. 5-17 General scheme for a standard RL strategy.. 75
Fig. 5-18 State machine implemented in the master system. 78
Fig. 6-1 Printing trajectory projected on a curved surface. 79
Fig. 6-2 Behavior of each joint qi for the different inverse kinematic solutions. 80
Fig. 6-3 The IK solutions available for a printing trajectory where the tracks for Joint3 and Joint4 are shown. 81
Fig. 6-4 RL strategy learned by the agent to evade the obstacles: a plane at $z = 30mm$, and a sphere inserted as an obstacle.. 82
Fig. 6-5 Behavior of the tracks for Joint3 and Joint4 for the available IK solutions and their relationship. 83
Fig. 6-6 IK1 solution chosen by the RL agent where an obstacle is blocking IK2, and a second obstacle is blocking IK7. 83
Fig. 6-7 a) Block diagram of the complete system and b) the actual UR3 with Flexion extruder attached. 84
Fig. 6-8 Actual system implemented in the laboratory. 85
Fig. 6-9 Re-entrant pattern printed on a curved surface using UR3. 86
Fig. 6-10 Another example of conformal printing as a hexagonal lattice. 86

List of Tables

Table 5-1 Commands implements for data transfer between the master and slave...... 62
Table 5-2 Denavit-Hartenberg parameters for UR3/CB3.. 69

INDEX

Chapter 1 Introduction 14

1.1 Background and motivation 14

1.1.1 Curved conformal 3D printing 14

1.2 Research aims and objectives 16

1.3 Overview 17

Chapter 2 Review of literature 18

2.1 Non-Planar Path Planning for Manufacturing 18

2.2 Non-Planar Path Planning for 3D Printing 20

2.3 Robotic Additive Manufacturing 24

2.3.1 Path planning for robotic 3D printing 25

2.3.2 Motion planning for 3D printing 25

2.4 Summary 26

Chapter 3 Trajectory planning for conformal printing on non-planar analytical surfaces 27

3.1 Generation of printing patterns from scratch as those used for lattices of metamaterials. 27

3.2 Generating a sinusoidal pattern. 28

3.3 Zig-zag pattern on a loft surface. 32

3.3.1 Printing a geometry that can be bent with conductive material to sense the ratio of blending. 35

3.3.2 Printing the same geometry that can be bent but embedding conductive wire. 37

3.3.3 Design and manufacturing of a multilayer printed spiral winding inductor. 41

3.3.4 Touch sensor using conductive thread and filament TPU. 44

Chapter 4 Trajectory planning for conformal printing on tessellated curved surfaces ... 46

4.1 Algorithm to project a trajectory on a non-planar triangular mesh surface. 46

4.2 Generation of printing trajectories from a collection of data located inside a file. . 48

4.2.1 Extracting information inside a copper Gerber file. 48

4.2.2 Algorithm to generate a trajectory from the information contained in a Gerber copper file (PCB). 49

4.2.3 Printing trajectories from the Gerber copper file of a circuit. 51

4.3 Lindenmayer system (L-system) for the methodological generation of printing patterns. 53

4.4 Conclusions 54

Chapter 5 Multi-axis system for 3D printing 56

5.1. Complete hardware system for 3D printing 56

5.1.1. The complete filament extruder mechanism. 57

5.1.2. Arduino-base microcontroller system 57

5.1.2.1. Ramps as a driver shield 58

5.1.2.2. Thermistor as a temperature sensor 58

5.1.2.3 Ceramic heater 60

5.1.2.4 Extrusion of material through the stepper motor 61

5.1.2.5 Programming the Arduino 61

5.1.2.5.1 Protocol of communication between the master computer and the slave Arduino-base board 62

5.1.2.5.2 Temperature control using a PID controller 62

5.1.3 UR3/CB3 Robot 63

5.1.3.1 Setting up the tool 64

5.1.3.2 Work object (build platform) configuration 66

5.1.3.3 Mathematical model of the UR3/CB3 69

5.2 Complete software system for 3D printing 70

5.2.1 Software that is executed before the printing task. 71

5.2.1.1 Generation of the printing trajectories and their projection (path planning). 71

5.2.1.2 Calculation of the transformation matrices. 72

5.2.1.3 Inverse kinematics for the manipulator's arm UR3 73

5.2.1.4 Application of a Reinforcement Learning (RL) strategy for optimization. 73

5.2.1.5 Defining the reward functions 75

5.2.2 Software during printing task 76

5.2.2.1 State machine to control the system 77

Chapter 6 Results obtained 79

6.1 Validation of the strategy for the conformal deposition of fused filament 79

6.2 RL agent solution with no obstacles 79

6.3 Obstacles avoidance 80

6.4 Using a manipulator arm for printing a lattice sample 83

6.4.1 Experimental setup 84

6.5 Discussion 86

Chapter 7 Conclusions 88

Chapter 1 Introduction

1.1 Background and motivation

Traditional industrial robots are expensive heavy articulated links in motion that have been used for the execution of repetitive tasks as they are surrounded by safety fencing. The complete analysis of their kinematics and dynamics may be a disquieting and complex task as well as their time-consuming programming through the teach pendant. Today, the availability of collaborative robots (cobots) that can safely interact with humans is increasingly popular. Cobots have revolutionized the original concept of industrial robots since they are provided with sensors to detect minimal collision and automatically stop their movements, they may be controlled wireless and work with third-party solutions adding extra functionality.

Some of the inherited and persistent challenges from traditional industrial robots are those related to path and motion planning, as well as the avoidance of obstacles. The development of new software tools may make it even easier the incorporation of cobots into the industry for the execution of complex trajectories while taking smart decisions. The time and effort for programming the robot should be reduced drastically when new software tools are available adding value to smart manufacturing processes such as additive manufacturing.

1.1.1 Curved conformal 3D printing

The process of additive manufacturing (AM) or 3D printing consists of the deposition of material on planar layers (e.g., XY) stacked along an axis (e.g., Z-axis) based on a digital model. One of the most used AM technologies is fused deposition modeling (FDM) due to its reliability and simple process (Hwang et al., 2015), but its constraints in the movement of the cartesian system produce limitedly functional prototypes. New strategies for freeform manufacturing have reached the stacking of planar layers that grows up in different directions (multi-axis), and curved trajectories (curved layers) for the deposition of material. These advantages have shown several benefits such as the 1) improvement of the finished surface, 2) improvement of mechanical performance, 3) achievement of specific functional properties and 4) the reduction of support structures, material, and printing time.

Conformal deposition of material, in the context of AM, has emerged as an attractive field for the achievement of specific functional properties of the manufactured object. Conformal printing is a variant of curved printing since it consists of the deposition of material on a freeform surface using a material extrusion process. The conformal deposition of conductive materials for printed electronics and the manufacturing of metamaterials to enhance mechanical properties are some examples of the applicability of conformal printing.

1.1.2 Applicability of curved conformal printing

In literature, the zig-zag pattern is mostly used for the implementation and validation of curved 3D printing by trying to smooth the surface, for instance. However, complex patterns such as those used for metamaterials (lattices) are more like repeating patterns connected in a way that the mechanical properties of the printed object may be tuned to achieve a range of responses. Similarly, in the context of Printed Electronics (PE), the deposition of conductive ink tracks needs to follow specific continuous trajectories that connect different electronic devices. In general, the curved additive manufacturing process would require following continuous curved tridimensional printing trajectories in which conventional AM algorithms are inadequate for the deposition of material.

Conformal printing for metamaterials and printed electronics require the generation of complex trajectories but also a system with a higher degree of freedom. As a solution, using a manipulator's arm allows further freedom of the nozzle’s movement, but several issues need to be considered and solved making the task sophisticated and challenging. Considerations concerning the process parameters such as deposition speed and extrusion rate must be satisfied. At the same time, there are relevant process planning considerations such as the orientation of the nozzle, the slicing process, and the toolpath that defines the printing quality and functionality of the geometry. Collision avoidance, singularities, and synchronization between the movement of the robot and the extrusion process are the main problems to be solved for conformal printing using robotics since available algorithms do not contemplate the generation of code compatible with the robot language. Thus, important aspects need to be solved using robotics for conformal printing, such as the need for continuous-as-possible printing paths (path planning), proper position and orientation of nozzle avoiding obstacles and singularities (motion planning), and the synchronization between the speed of the end effector (extruder or build platform) and the flow rate (extrusion) of material.

1.2 Research aims and objectives

The main objective of this work is to propose a framework solution to the challenges mentioned previously to satisfy the important aspects when the deposition of material on curved surfaces (analytical and tessellated) is required following complex trajectories. The scope of the work includes the process planning for curved conformal 3D printing generating the machine-code for a conventional 3D printer (2.5 degrees of freedom) and a manipulator arm (more than 2.5 degrees of freedom).

The specific objectives of the present work are:

1. Develop an algorithm for the generation of complex conformal printing trajectories on non-planar surfaces (both analytical and tessellated surfaces) which satisfy the feature of continuous-as-possible trajectories.
2. Generation of the corresponding machine-code for moving the extruder using a cartesian system (conventional 3D printer).
3. Propose a complete framework system for the achievement of FDM using a manipulator's arm.
4. Develop collision-free path planning and motion planning for curved printing using a manipulator arm by considering the robot constraints.
5. Develop a strategy for the achievement of the synchronization between the movement of the end effector and the requirements of manufacturing.

This work will demonstrate the relevance of how having new software tools, the incorporation of a robot framework, and complex conformal trajectories to the smart additive manufacturing techniques can effectively speed up making the process easier.

1.3 Overview

Chapter 2 presents a review of the literature on curved 3D printing, its main benefits, and its main challenges when using a multi-axis system such as a manipulator robot. Chapter 3 explains the mathematical background for the generation from scratch of complex trajectories using parametric equations on analytical surfaces. The validation is implemented with the manufacturing of functional (electrical) devices that contain a conductive material, and the manufacturing of lattices as metamaterials. Chapter 4 contains the developed algorithm to transfer complex parametric trajectories to non-planar tessellated surfaces and some examples for its validation. Chapter 5 explains the complete printing system implemented using a manipulator robot for the conformal deposition of material on curved surfaces having the nozzle normal oriented to the surface. The path and motion planning strategy are explained for the obstacle avoidance and achievements of the manufacturing requirements. Chapter 6 contains the results obtained during the execution of the system. Finally, chapter 7 contains the conclusions and future work of the research project.

Chapter 2 Review of literature

The review of literature includes the robust solutions to path planning for manufacturing processes for classical problems (e.g., machining, painting, welding, and polishing) on curved surfaces as the basis of subsequent methods adapted for curved printing. The new strategies that have arisen for curved 3D printing are also included, such as those focused on multi-axis systems and robotics.

2.1 Non-Planar Path Planning for Manufacturing

According to (H. Chen et al., 2009), multi-axis tool path planning methods can be categorized into two groups: those based on parametric CAD models (analytical geometry) and those based on tessellated CAD models (faceted geometry). A parametric surface, such as ruled or loft surfaces, is defined by two or more path curves on opposite sides of the surface joined by straight lines or a loft surface. Alternatively, faceted geometry (e.g., STL files) has the advantage of being defined with simply obtained data, such as vertices and normal vectors without any parametric structure and differential attributions (J. Zhao et al., 2015). According to (T. Chen & Shi, 2008) and (Lasemi et al., 2012), the three popular methods for tool path generation in multi-axis machining are iso-parametric (Broomhead & Edkins, 1986; Kuragano, 1992; Loney & Ozsoy, 1987), iso-planar (Bobrow, 1985; Y. D. Chen et al., 1993; Y. Huang & Oliver, 1994), and iso-scallop height (Lin & Koren, 1996; Sarma & Dutta, 1997; Suresh & Yang, 1994). In the iso-parametric method, the contact points along the desired path are generated directly by keeping one of the two parameters as a constant (e.g., u) and varying the other (e.g., w) (Lasemi et al., 2012). The iso-parametric method may not be suitable for a compound surface consisting of a collection of surface patches (Choi & Jerard, 1998), and its main disadvantage is that scallop heights are not constants due to differences between the Cartesian and parametric space (Mladenovic et al., 2020; Mladenović et al., 2015). Iso-planar (Cartesian) tool paths are generated by intersecting a surface $S(u,w)$ with parallel planes $(P_1, P_2, \ldots, P_n)$ in Cartesian space as shown in **Figure 2-1**. This method can be used for compound surfaces, trimmed surfaces, and tessellated surfaces (Lasemi et al., 2012). One advantage of this method is its uniform interval between adjacent tool paths in the Euclidean space. The third method, named iso-scallop, is an improvement of isometric and iso-planar methods, according to (Mladenović et al., 2015), and it generates subsequent tool paths based on the known preceding paths. These robust path planning methods are widely used for traditional industrial tasks (e.g., machining, painting, welding, polishing, and glue dispensing) and have established the basis for consecutive methods for printing.

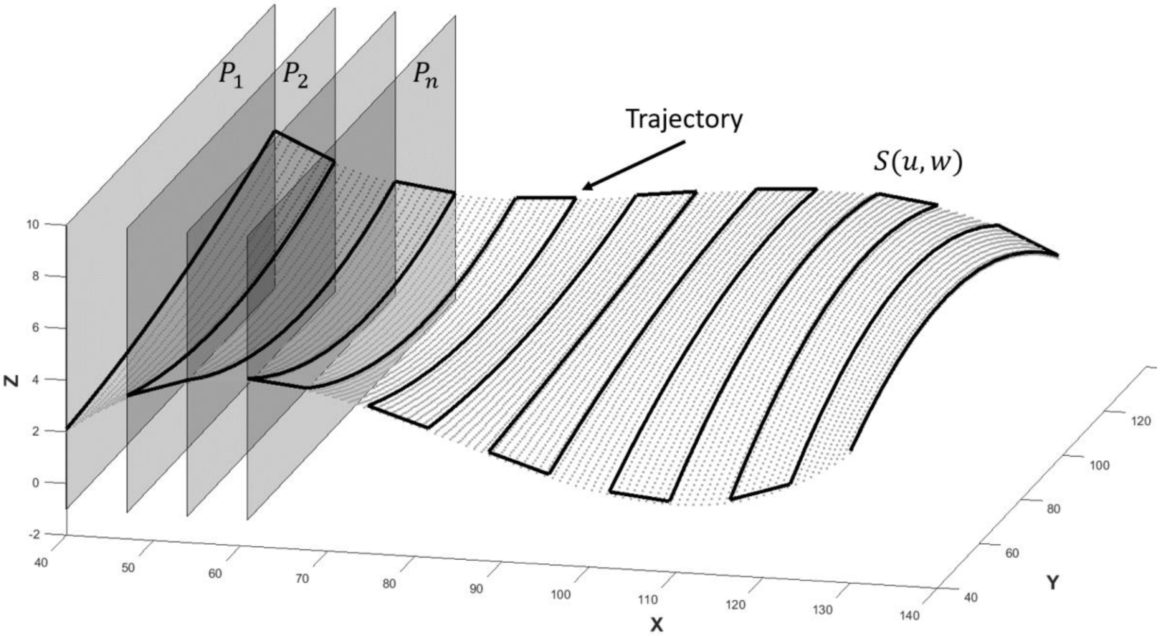


Fig. 2-1 Iso-planar (Cartesian) tool path generation.

Several methods for tool path planning on triangulated surfaces have also been explored. Some authors have proposed algorithms for triangular mesh surface parameterization methods by solving partial differential equations, as (M. Jin et al., 2018) explained. Chen et al. (H. Chen et al., 2003) explained a method based on the iso-planar using a tessellated CAD model, where a bounding box is defined, and the tool path is generated by cutting the bounding box along the top and front directions. Jun et al. (Jun et al., 2002) generated paths for machining from an STL by offsetting the polyhedral model and intersecting the offset surface with drive planes as shown in **Figure 2.2**. Similarly, (Lauwers et al., 2003; Mineo et al., 2017) developed a 5-axis milling tool path generation based on faceted models consisting of contact points at the intersection of a set of parallel planes and the edges of the triangles. Lauwers et al. (Lauwers et al., 2003) calculated the normal vector at a given point as the average of the neighborhood triangles' normal.

The strategies for the non-planar path planning for subtractive manufacturing give a start point for those strategies used for additive manufacturing, but most of them need to be adapted. The path planning for subtractive manufacturing does not consider an appropriate order of sub-paths since this does not affect the result. Similarly, the need for continuous path planning is not a requirement nor is the aim of less head up and down times. On the other hand, 3D slicing and path sequence strategy for AM are crucial for the proper achievement of the manufacturing requirements that affect the shape and performance of the printed parts.

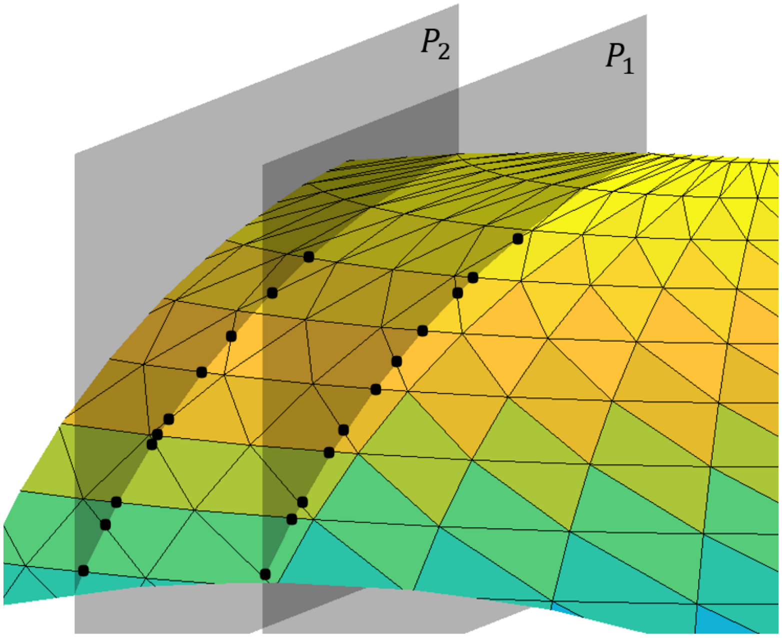


Fig. 2-2 Points at the intersection between edges of triangles and orthogonal planes.

2.2 Non-Planar Path Planning for 3D Printing

The term curved layer printing and its variants mainly refer to slicing a model using curved layers instead of the traditional planar layer slicing or a combination of planar and curved. A very complete review of planar and non-planar slicing methods and path planning for AM is presented by (D. Zhao & Guo, 2020). Different approaches are also found in the literature related to conformal printing on non-planar surfaces that are not commonly associated with the curved slicing model but are included here due to their importance in this study. The term CLFDM was first introduced by (Chakraborty et al., 2008) who formulated a theoretical method, mainly based on CNC traditional concepts, for the manufacturing of thin curved shells to improve the mechanical properties and reduce the stair-step effect. Their work was based on employing longer-length tool paths, focused on the proper orientation of the filament and appropriate bonding between adjacent filaments. Their formulation used a parametric surface, calculated its partial derivative of it to obtain the normal vector, and then generated an offset surface (**Figure 2-3**). The concept of CLFDM was experimentally reported by several researchers, such as (Diegel et al., 2011; B. Huang et al., 2008; S. Singamneni & Diegel, 2010). They generated the path planning (flat layers) to produce a mandrel as a support structure where the curved layers were later deposited, following the contour of the part. Later, (Sarat Singamneni et al., 2012) improved their algorithm of the cross-product of four vectors by considering a vertical plane passing through three consecutive surface points. Even though all their contributions laid the foundations of experimental CLFDM, some important details about the implemented algorithms were skipped. They studied the generation/selection of data points to produce the offset curved layer directly from the G-code or M-code generated by CAM software, and as a result, the printing trajectory was limited to those points. Some authors have developed curved layer slicing by modeling and fitting the surface using B-spline, such as (Y. Jin et al., 2017), who fitted an STL mesh surface with a B-spline surface with two independent parameters (u and v). They modified the original tessellated surface to reduce the number of triangles of the STL file and then fitted the surface. The first printing path (the author recommends it to be along one of the edges of the design) defines the next paths, which are generated by a certain offset (equidistant). They reported some limitations in the processing of the part surface due to the CAD and CAM software used. Patel et al. (Patel et al., 2015) optimized the number of curved layers needed for printing by preserving the critical features. They modeled a B-spline surface using selected critical points and generated curved offset layers optimized by the application of genetic algorithms and surface-surface intersection. Their results included simulations but a nonphysical implementation. Allen and Trask (Allen & Trask, 2015) used a delta configuration system and generated the printing path of a surface or skin defined mathematically and a core component (infill pattern) as a matter of contrasting distinct structural or physical functions, as (Llewellyn-Jones et al., 2016) later also demonstrated by producing models with aesthetic and structural properties. The algorithm consisted of converting the analytical surface in a grid XY following the points in order and calculating z for dynamic movements. McCaw and Cuan-Urquizo (McCaw & Cuan-Urquizo, 2018) presented the procedure to fabricate nonplanar lattice shells on non-planar equation-defined surfaces (parametric Bèzier surfaces of arbitrary order), whereas (Cuan-Urquizo et al., 2019) generated and fabricated a lattice using rectangular equations and studied the mechanical behavior when force is applied. McCaw and Cuan-Urquizo (McCaw & Cuan-Urquizo, 2020) presented a mathematical approach to parametrize lattices onto Bèzier surfaces to fabricate non-planar chirality lattices and studied them under cyclic loading. Conformal printing has emerged as a process to deposit silver inks on curvilinear surfaces to create conductive paths (Adams et al., 2011); however, some recent studies were published about path planning for conformal 3D printing using FDM. (Shembekar et al., 2019) proposed an algorithm for conformal printing using non-planar layers and evaluated the differences in roughness between a surface finish when printed using planar layer slicing and the proposed algorithm. The algorithm aims at collision-free trajectory planning using a projection method (**Figure 2-4**): (1) a grid is created on the XY plane (0.5 mm spacing); (2) vertices of each triangle are projected to the XY plane; (3) specific points of the grid belong to a particular triangle; (4) the equation of the plane of the triangle is calculated from three vertices; and then (5) the z value for these points inside the triangle is calculated and mapped back to the non-planar surface. As an evaluation, a zigzag pattern at two different angles is used to improve the finishing of the surface. (Alkadi et al., 2020) proposed an algorithm to locate conformally one tessellated structure onto a second tessellated surface (substrate). The algorithm achieves the following: (1) it generates a curved slicing surface by offsetting the top of the substrate; (2) it obtains the boundaries of the pattern to be printed by the intersection of the structure and the slicing surface; and (3) 2D printing patterns are projected to create 3D patterns. To achieve conformal trajectories, this algorithm has the restriction that the bottom of the 3D structure must fit the freeform substrate, and in the case of a mismatch, the free spaces are filled to connect both structures (**Figure 2-5**). The algorithm outputs the G-code for 3D printing. A different approach for printing quality improvement was proposed by (Ahlers et al., 2019), who developed an algorithm for planar and non-planar slicing. Their main contribution is the detection of the parts suitable to be printed using non-planar slicing assuring collision-free toolpaths, using a simplified printhead model defined by the maximum nonplanar angle and the maximum nonplanar height. The printing trajectories presented are focused to achieve smooth surfaces (zigzag pattern). Feng et al. (Feng & Cui, 2021) implemented a five-axis machine (a delta printer plus a platform rotating) and proposed an algorithm for curved layer material extrusion. Their main contribution is the reduction of the material used for the mandrel to achieve conformal curved printing; hence, the printing time is also reduced. They generated a conformal surface offset and a toolpath using the geodesic distance as the shortest zigzag along the facet edges of the STL file. The path planning consisted in equidistantly offsetting the starting curves.

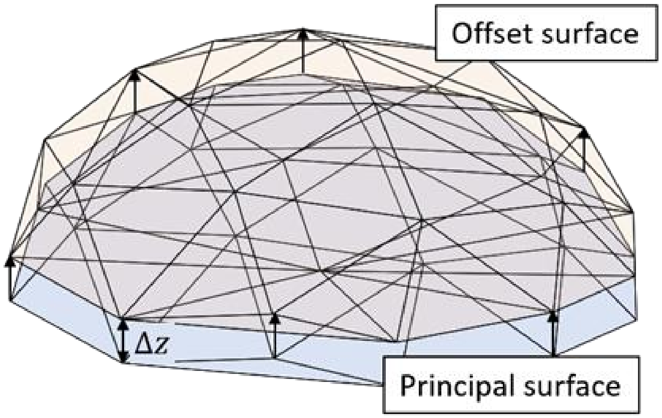


Fig. 2-3 Generation of curved slicing layers by offsetting the z-value of vertices of the upper surface.

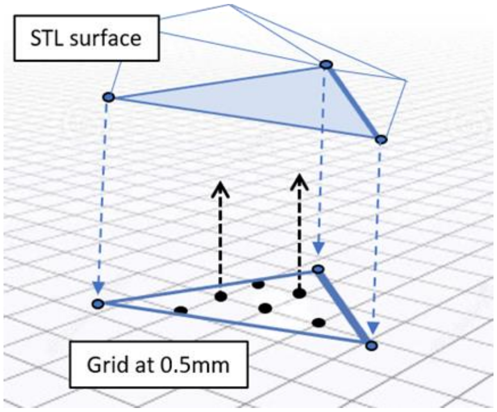


Fig. 2-4 Projection of zig-zag trajectory for surface finishing (smooth): vertices of triangles of tessellated target surface are projected to a spacing grid XY and the grid points, which belong to a certain triangle, are projected back to the target surface.

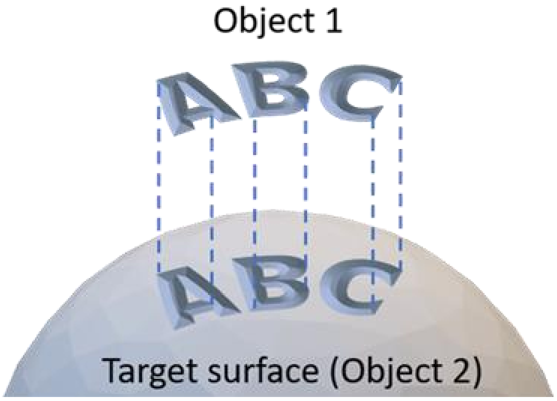


Fig. 2-5 A tessellated object_1 is conformally attached to a second tessellated object_2

The review of the strategies used for non-planar path planning for 3D printing shows, to the best of our knowledge, that most of them are focused to improve the surface finish and mechanical performance. The most used printing paths for these purposes are limited to zigzag and circular patterns. Similarly, the printing paths consist of equidistantly offsetting starting curves which restricts the points to define the printing trajectory. A limited number of works were focused on the generation of more complex trajectories, such as those needed for the manufacturing of lattices or specific paths for the deposition of fibers or conductive materials. Complex trajectories or patterns, such as sinusoidal lattices, were reported, and these were printed on analytical surfaces (mathematically parametrized) but not using tessellated surfaces. Complex free forms, such as those encountered in biomedical applications (normally obtained from scanned data), may not be mathematically parametrized. Hence, approaches such as the one presented in this paper, gain relevance; any surface, parametrizable or not, could be modeled using a tessellated surface.

2.3 Robotic Additive Manufacturing

The different strategies for curved 3D printing need a multi-axis system to be implemented experimentally. Some of them have been achieved using conventional cartesian or delta 3D printers where the nozzle is (vertically) fixed-oriented and dynamically moved along the z-axis. In this case, the main constraint is the printing of geometries without significant steep slopes to avoid the collision between the nozzle and the object that is being printed. Smaller extruders and/or larger nozzles are frequently suggested and replaced as a solution. Similarly, the use of robotics as a multi-axis system has shown promising results even with the mathematical background needed for its implementation and high cost.

2.3.1 Path planning for robotic 3D printing

Path planning in robotics involves finding the optimal path between an initial point and a target point. For Additive Manufacturing (AM), the printing path planning involves slicing, outlining, and infilling curves. Zhao and Guo reviewed the literature on slicing and path planning for AM (D. Zhao & Guo, 2020). From there, the continuous printing trajectories stand out, such as the connected Fermat spirals (H. Zhao et al., 2016) incorporated as space-filling curves to avoid discontinuities in AM using robotics. As an example, (Xie et al., 2022) proposed an algorithm for curved layer decomposition of an arbitrary freeform solid model, based on the use of ellipsoidal surfaces and the skeletonization technique. This algorithm was developed for complex features and non-zero genus numbers. On the other hand, analytical modeling such as following directly parametric equations to generate printing contours has been found in the literature. (Yigit & Lazoglu, 2019) proposed a non-planar helical slicing algorithm to generate a helical continuous three-dimensional printing path eliminating the non-extrusion movements of the extruder and reducing the defects. The algorithm is designed for the manufacturing of freeform single-walled shell type, genus zero models with no infill, and minor overhangs. Curved layers for AM are frequently generated by applying the iso-parametric strategies used widely for freeform machining on parametric surfaces (L. Chen et al., 2019; G. Zhao et al., 2019). Zhao et al. (G. Zhao et al., 2018) extracted curved layers from tessellated geometries using a cylinder surface to slice the model. (Shembekar et al., 2018) uses a simple zig-zag pattern to generate a collision-free trajectory planning for conformal AM using non-planar layers and the projection of the pattern on a tessellated surface.

2.3.2 Motion planning for 3D printing

Motion planning is defined as the capability of a robot to taking its own decisions regarding its displacements under certain circumstances such as static or dynamic scenarios using optimal strategies to achieve the target position from a starting point

(Zhou et al., 2021, 2022). The motion planning algorithms may be classified into 1) traditional algorithms: graph-based search algorithms, sampling-based algorithms (e.g., Probabilistic Roadmap Methods, Rapidly Exploring Random Trees), interpolating curve algorithms (e.g., Polynomial, Bezier, and Splines), and reaction-based algorithms (e.g., artificial potential field), and 2) Machine-Learning (ML) algorithms: supervised, optimal value Reinforcement Learning and policy value Reinforcement Learning. The basic motion planning problem may be solved by 1) representing the scenario (obstacles and free space) and 2) searching for a path that connects the starting point and the endpoint without collisions.

(Xie et al., 2020) proposed the iso-cusp height expansion method that uses a potential field-based algorithm to generate printing trajectories by subdividing the analytical surface to be printed in sections according to an optimal orientation of the extruder. Their main contribution is the generation of a kinematically efficient path such that the printing time was also reduced. Since their work is based on their previous work (Hu & Tang, 2016) focused on machining tasks, the continuous printing trajectories are not achieved since the nozzle needs to travel between the different sections; on the other hand, collisions are avoided by only limiting a range of the movement of joints. (Shembekar et al., 2019) recall the requirements of constraints of speed and collision avoidance and suggest the calculation of the inverse kinematics (IK) and graph-based search for avoiding collisions and solving inverse Jacobian and integrating over the domain to assure proper printing speed.

2.4 Summary

The use of an arm manipulator for the deposition of the material using curved trajectories rather than planar has drawn attention as a solution for the achievement of a higher degree of freedom conditions. The traditional path planning and motion planning problems in robotics need extra considerations and requirements to be satisfied for 3D printing and quality achievement. However, obstacle avoidance requires an exhausting analysis due to the high dimensionality. Here a solution is proposed using Reinforcement Learning which can be accurately implemented in problems with high dimensionality and uncertain scenarios, and where the number of actions increases exponentially with the number of degrees of freedom.

Chapter 3 Trajectory planning for conformal printing on non-planar analytical surfaces

Geometric modeling techniques such as the mathematical representation of parametric curves and surfaces are the basis employed in CAD and they can be used to generate conformal parametric trajectories for printing. A very popular curve and surface format is the parametric representation. Using the processes of extrusion, blend (or loft), revolve, or sweet, a surface or solid can be generated as an example shown in **Figure 3-1**.

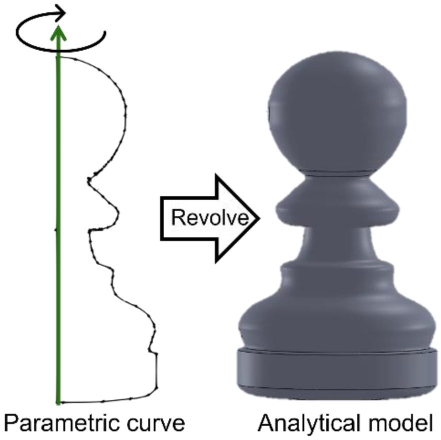


Fig. 3-1 A solid model is created by revolving a profile curve around an axis.

Complex printing trajectories for the conformal deposition of material can be generated using parametric equations. These complex patterns, as those used for lattices of metamaterials, can be generated methodologically from scratch using the analytical representation of the surface on which the paths will be located. Similarly, the printing trajectories can be generated from a collection of data located inside a file, such as the information contained in the Gerber files describing the conductive tracks of a printed circuit board (PCB) to be manufactured.

3.1 Generation of printing patterns from scratch as those used for lattices of metamaterials.

Traditional printing trajectories are 2D infill patterns that are selected from different 3D slicing software. For instance, some of the infill patterns found in Cura Software V4.5 are grid, triangles, cubic, octet, concentric, zigzag, cross, and gyroid, among others (**Figure 3-2**). Most of the patterns require the retraction of the filament at some point. It is not documented that the slicing software aims to reduce the number of retractions of the filament during the generation of the printing trajectories. On the other hand, patterns used for lattices are more like repeating patterns connected in a way that the mechanical properties of the printed object may be tuned to achieve a range of responses. Some of these mechanical properties are realized when a continuous printing trajectory is generated such as lattice-reinforcement. Some of these patterns found in lattices may be more complex than conventional infill patterns as it may be desired that they were generated using continuous trajectories such as the hexagon (re-entrant), Hilbert, and sinusoidal, among others.

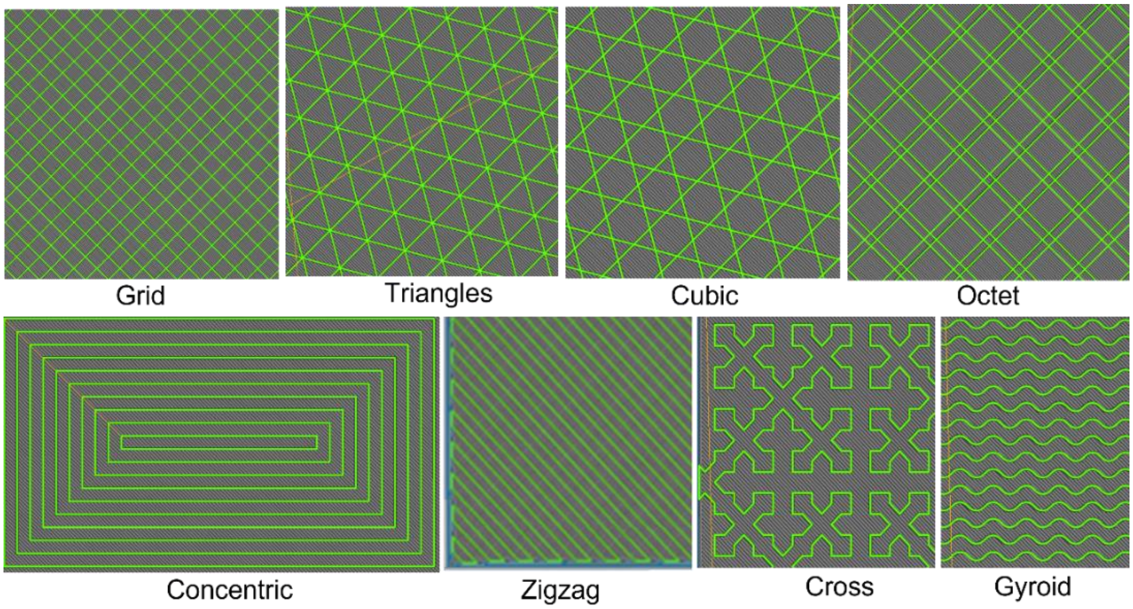


Fig. 3-2 Some infill patterns found in Cura V4.5 slicing software.

3.2 Generating a sinusoidal pattern.

One of the patterns used for metamaterials is the sinusoidal trajectory repeated along an axis and alternated on each layer to form a lattice. To generate such a sinusoidal lattice, first, a sine waveform is generated using the parametric equation of a cubic curve as defined in $(3-1)$ through $(3-3)$, where the points $P_1 - P_4$ are located in a way to define the concordance of the sinusoidal waves in the X-axis and Y-axis (**Figure 3-3**). Having generated a complete sinusoidal cycle, some parameters are user-defined as the period of the waveform, the number of waveforms by row, the offset to locate the initial printing position of lattice on the build platform, and the number of layers to form the complete lattice as the example indicated in **Figure 3-4** where $period = 10$ in millimeters.

The printing trajectory is defined to be continuous from row to row and layer to layer following the direction as indicated in **Figure 3-5.**

$$P(u) = \boldsymbol{UN^S G^S} = [u^3 \quad u^2 \quad u \quad 1] \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} \qquad (3-1)$$

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = \begin{bmatrix} 0 & period/2 & 0 \\ period/2 & period & 0 \\ period/2 & 0 & 0 \\ period & period/2 & 0 \end{bmatrix} \qquad (3-2)$$

$$[\mathrm{x}, \mathrm{y}, \mathrm{z}] = [u^3 \quad u^2 \quad u \quad 1] \begin{bmatrix} 0 & 3\,period & 0 \\ -3\,period/2 & -9\,period/2 & 0 \\ 3\,period/2 & 6\,period/2 & 0 \\ 0 & period/2 & 0 \end{bmatrix} \qquad (3-3)$$

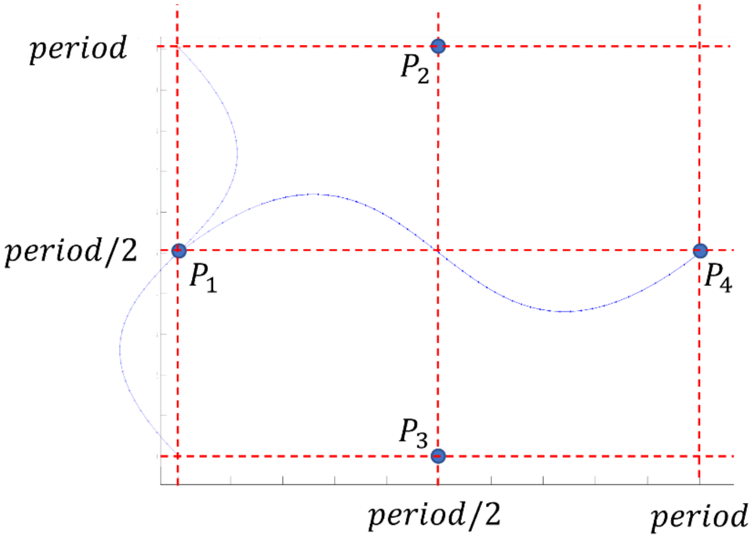


Fig. 3-3 Points $P_1 - P_4$ that define the sinusoidal waveform in X-axis and Y-axis.

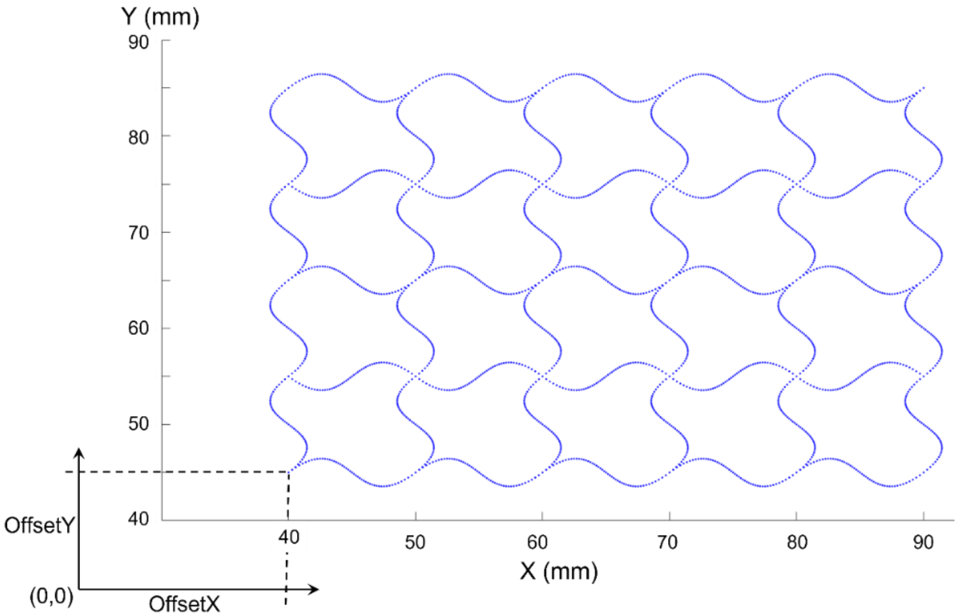


Fig. 3-4 User-defined parameters for the complete sinusoidal pattern for lattice.

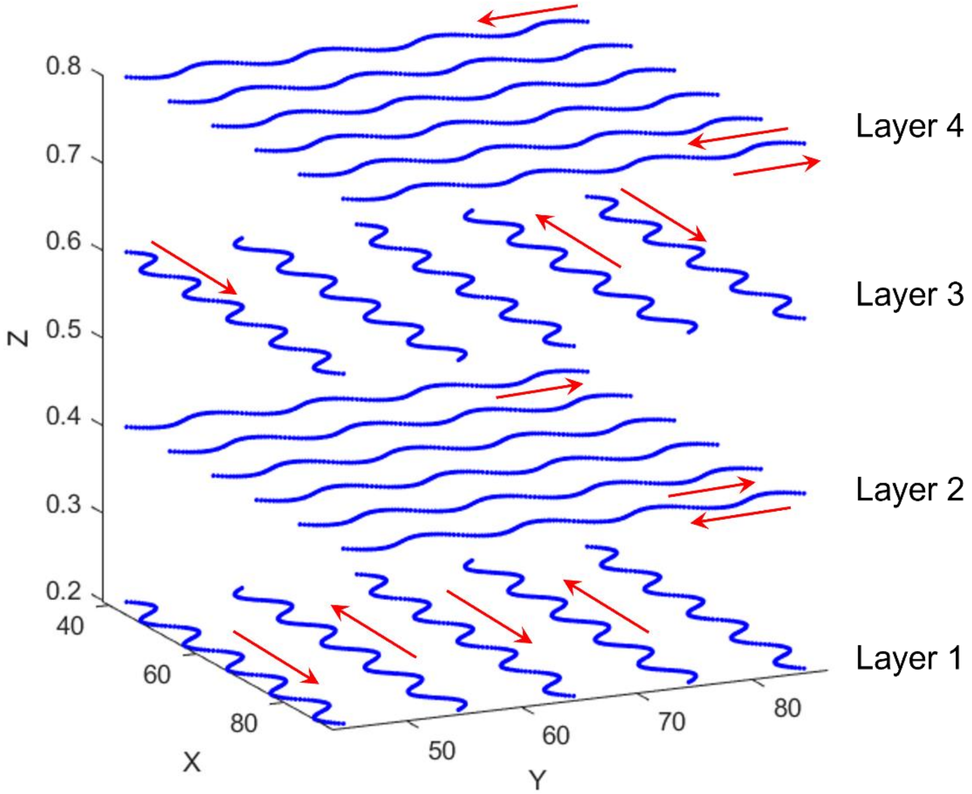


Fig. 3-5 Layers of the printing sinusoidal trajectory and the printing direction.

To generate a sinusoidal lattice mounted on an analytical surface, the values (x, y) of the sinusoidal waveform are translated to a new (x, y, z) value according to the analytical surface where those values are going to be mounted as represented in

Figure 3-6. For example, the previous sinusoidal lattice on the XY plane is going to be translated to cylindrical support whose parametric profile is given by $(3-4)$ through $(3-6)$ on plane YZ. The different views of the printing trajectories are shown in **Figure 3-7**. The printing layers are alternated as before and indicated in **Figure 3-8**

$$P(u) = \boldsymbol{U}\boldsymbol{N}^{S}\boldsymbol{G}^{S} = [u^2 \quad u \quad 1]\begin{bmatrix} 2 & -4 & 2 \\ -3 & 4 & -1 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} \qquad (3-4)$$

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & width/2 & height \\ 0 & width & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 23 & 5 \\ 0 & 46 & 0 \end{bmatrix} \qquad (3-5)$$

$$[\mathrm{x}, \mathrm{y}, \mathrm{z}] = [u^2 \quad u \quad 1]\begin{bmatrix} 0 & 0 & -20 \\ 0 & 46 & 20 \\ 0 & 0 & 0 \end{bmatrix} = [0,\ 46u,\ -20u^2 + 20u] \qquad (3-6)$$

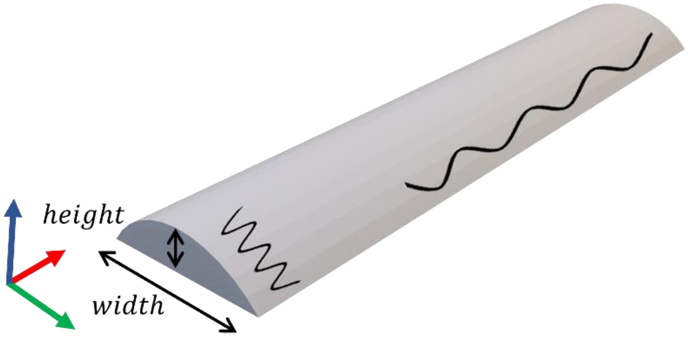


Fig. 3-6 Sinusoidal waveforms are translated to a cylindrical parametric surface defined using three points $P_1 - P_3$.

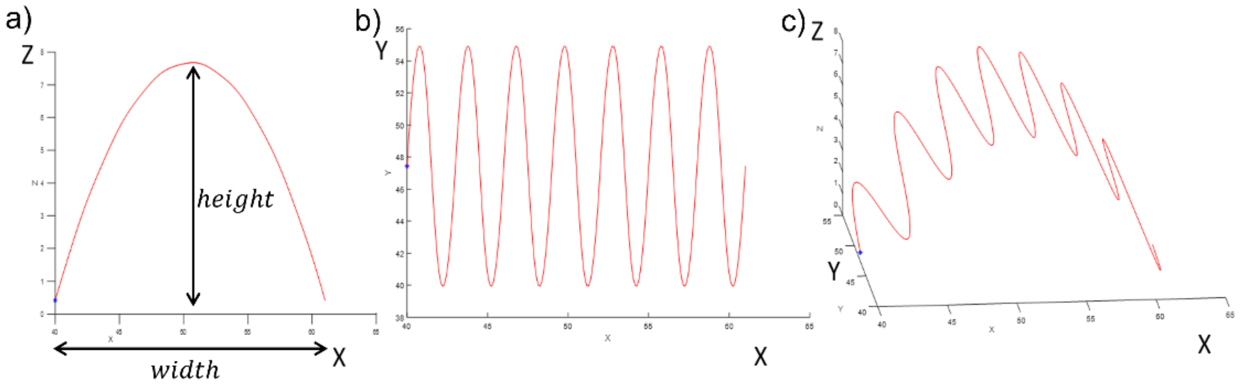


Fig. 3-7 a) Parametric profile of the support, b) sinusoidal waveform running along the X-axis, and c) sinusoidal translated to the parametric profile of the cylindrical section.

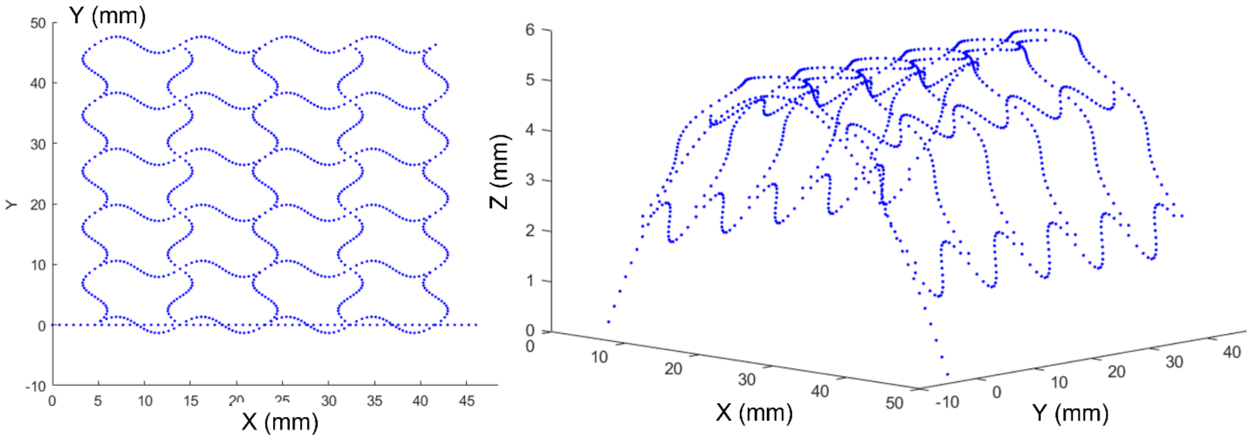


Fig. 3-8 View X-Y of sinusoidal lattice for printing (left), and tridimensional view of lattice translated to the parametric profile (right)

The experimental validation was executed using PLA of $1.75mm$, an Ender-3 printer, and a nozzle of $0.4mm$. **Figure 3-9** shows the steps followed to print conformally the sinusoidal trajectories on the section of the cylindrical parametric profile: first, the cylindrical support is planar sliced using the software Cura and printed as a conventional printing, then the mandrel is covered using painter tape to allow the detaching of the piece. The lattice printing trajectories are simulated and the G-code for the printer is generated.

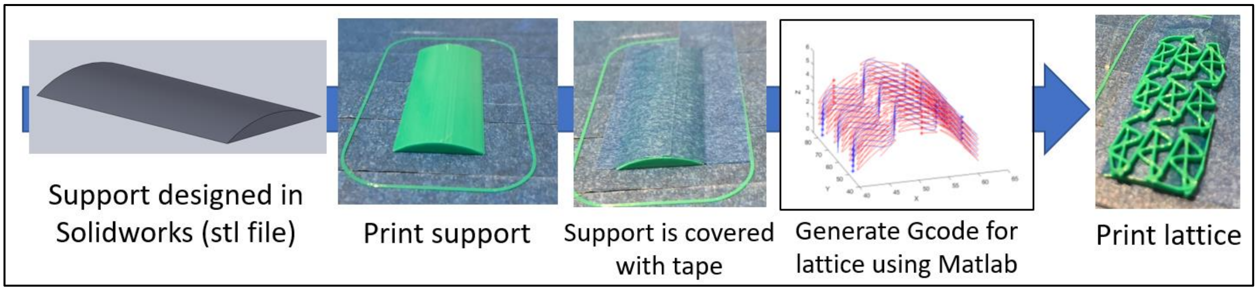

Support designed in Solidworks (stl file)
Print support
Support is covered with tape
Generate Gcode for lattice using Matlab
Print lattice

Fig. 3-9 Steps followed for the experimental validation of sinusoidal pattern on an analytical surface using a conventional Ender-3 printer.

3.3 Zig-zag pattern on a loft surface.

Any printing trajectory can be also generated on an arbitrary loft surface S defined as in $(3-7)$ and using three sketches' profiles $P(u), Q(u)$, and $R(u)$ as shown in **Figure 3-10**. The profiles $P(u), Q(u)$, and $R(u)$ are cubic parametric equations obtained from four points as defined in $(3-8)$. The main features of the surface may be user-defined such as the offset to locate the lattice on the build platform (XY), the number of divisions for the parameters u and v, and the number of layers to be printed.

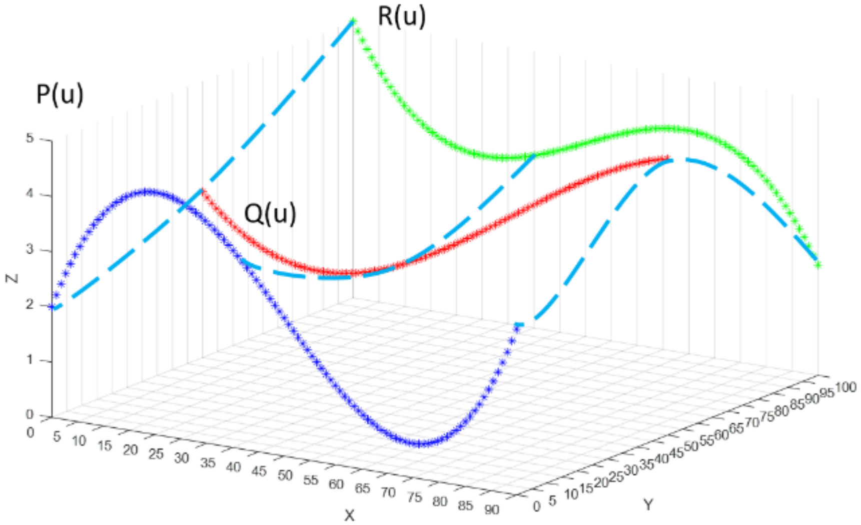


Fig. 3-10 Surface from three sketches profiles

$$S(u,w) = \begin{bmatrix} w^2 & w & 1 \end{bmatrix} \begin{bmatrix} 2 & -4 & 2 \\ -3 & 4 & -1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} P(u) \\ Q(u) \\ R(u) \end{bmatrix}, (u,w) \in [0,1] \times [0,1] \qquad (3-7)$$

$$P(u) = \boldsymbol{UA} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} -9/2 & 27/2 & -27/2 & 9/2 \\ 9 & -45/2 & 18 & -9/2 \\ -11/2 & 9 & -9/2 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} \qquad (3-8)$$

The experimental validation was executed using a zigzag trajectory on a loft surface defined by the following profiles in millimeters chosen in purpose to avoid the collision of a nozzle with the build platform:

$$P(u){:}\, P_1 = (0,0,2), P_2 = (30,0,4), P_3 = (60,0,1), P_4 = (90,0,3)$$

$$Q(u){:}\, P_1 = (0,50,3), P_2 = (30,50,2), P_3 = (60,50,1), P_4 = (90,50,5)$$

$$R(u){:}\, P_1 = (0,100,5), P_2 = (30,100,3), P_3 = (60,100,1), P_4 = (90,100,2)$$

It used a filament of PLA of $1.75mm$, an Ender-3 printer, and a nozzle of $0.4mm$. **Figure 3-11** shows the steps followed to print conformally a zigzag trajectory on this analytical surface. **Figure 3-12** shows the results of the actual printing using a conventional 3D printer.

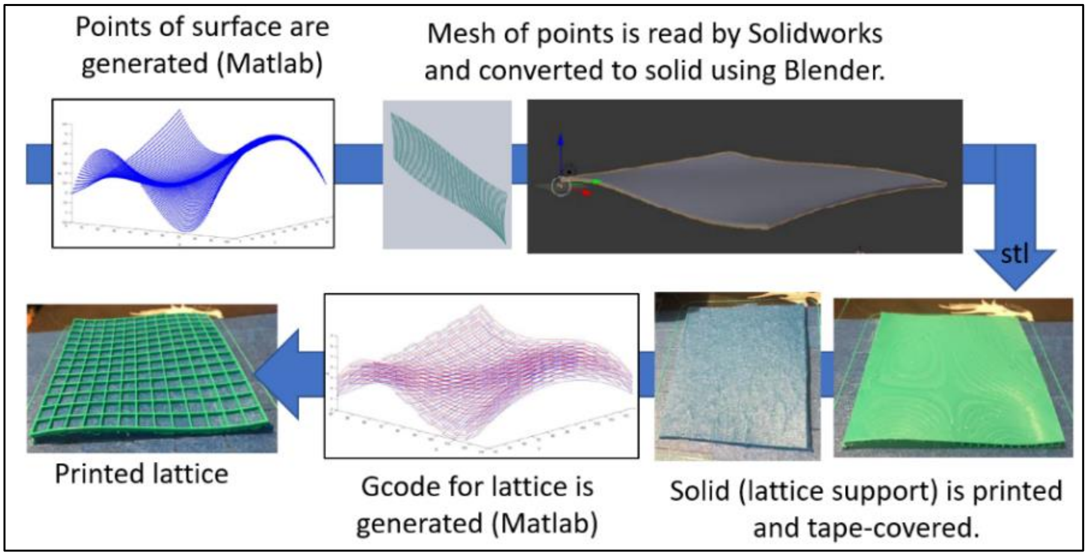

Points of surface are generated (Matlab)
Mesh of points is read by Solidworks and converted to solid using Blender.
stl
Solid (lattice support) is printed and tape-covered.
Gcode for lattice is generated (Matlab)
Printed lattice

Fig. 3-11 Steps for printing a lattice using a zigzag trajectory on an analytical surface.

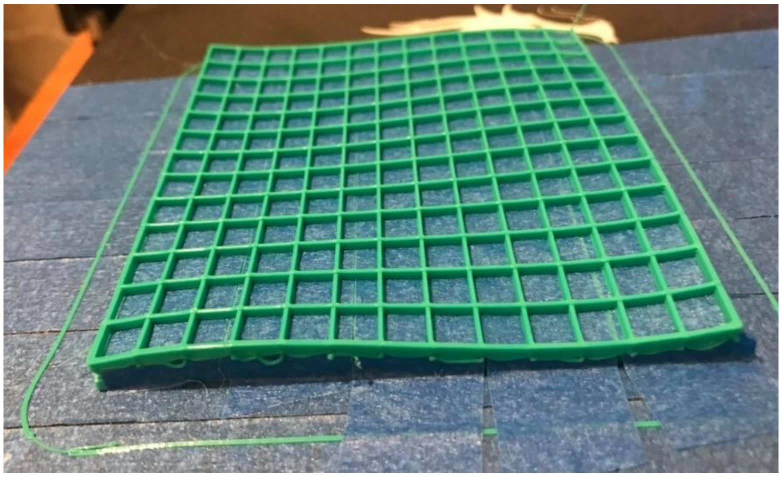


Fig. 3-12 Actual printed lattice on an analytical surface.

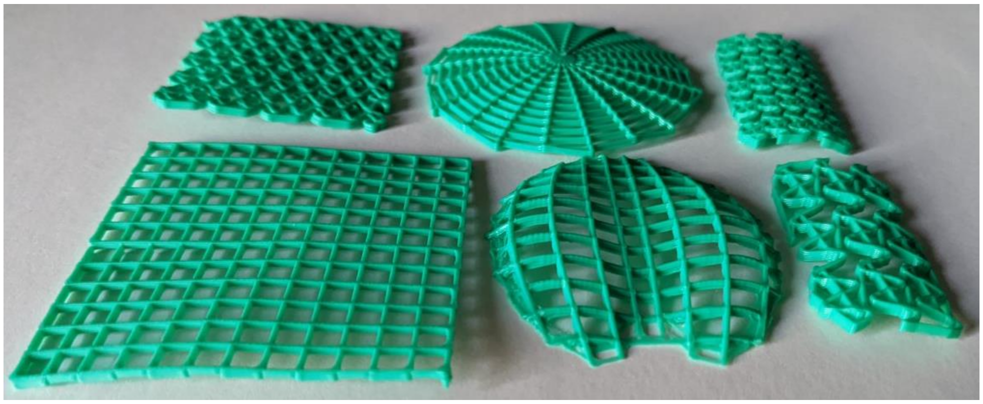


Fig. 3-13 Different trajectories generated on analytical surfaces using a conventional Ender-3 printer.

3.3.1 Printing a geometry that can be bent with conductive material to sense the ratio of blending.

For the experimental test of the conductive ink, a special bendable geometry was designed as shown in **Figure 3-14** and **Figure 3-15**, expecting a behavior as a flex sensor. The design has some incisions allowing the object can be bent, even though the geometry was printed as a planar object. By the deposition of conductive material following a continuous conductive trace along the geometry, the response of the functional device was expected to change proportionally to the bending ratio.

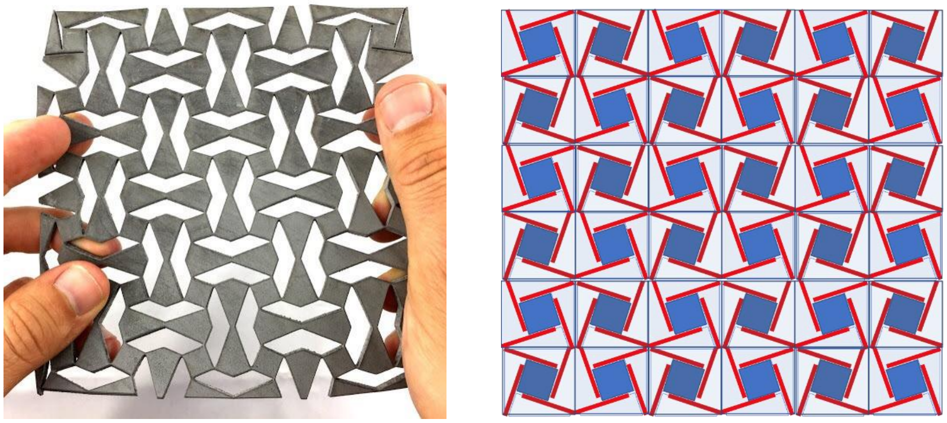


Fig. 3-14 Planar geometry with bending characteristics.

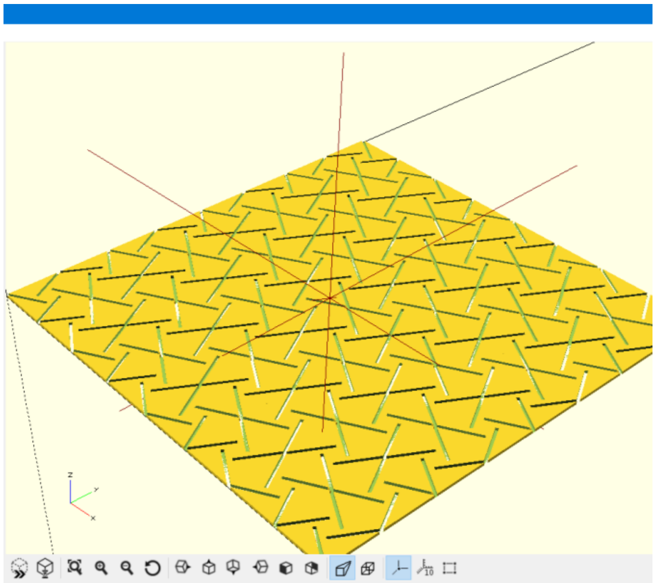


Fig. 3-15 Design of a bendable planar geometry.

Due to the pandemic, the lab equipment was not available. As a solution, the application of the conductive ink was made using simple tools and by hand to produce a sensor to measure the ratio of bending of the object. After printing the object, the first layer of conductive ink was applied and cured for 24hrs at ambient temperature, then a second layer was applied and cured again. The results are shown in **Figure 3-16**. Unfortunately, the conductive ink detached immediately after touching and bending the object as shown in **Figure 3-17**.

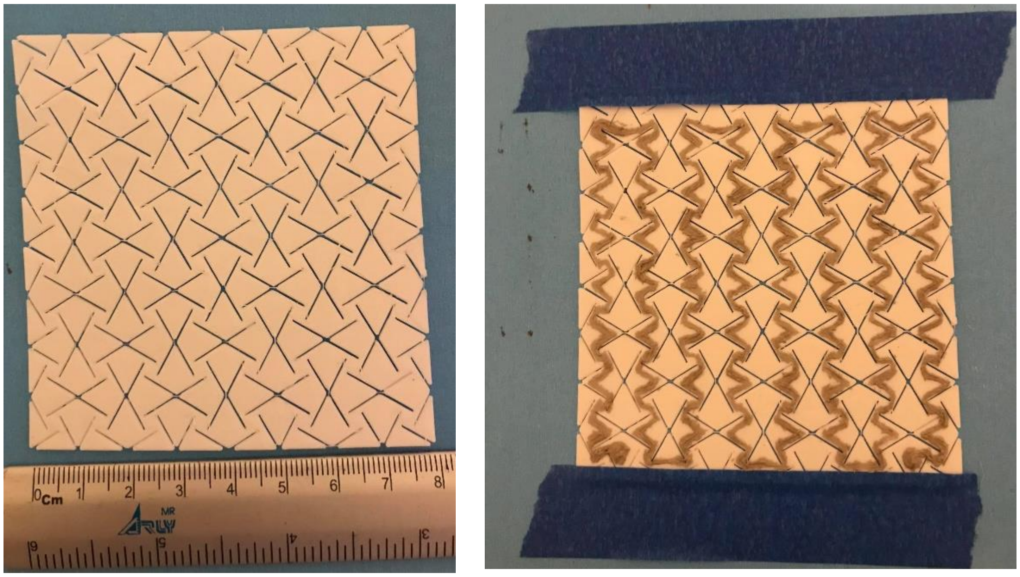


Fig. 3-16 Two layers of conductive ink were applied and cured for 24hrs.

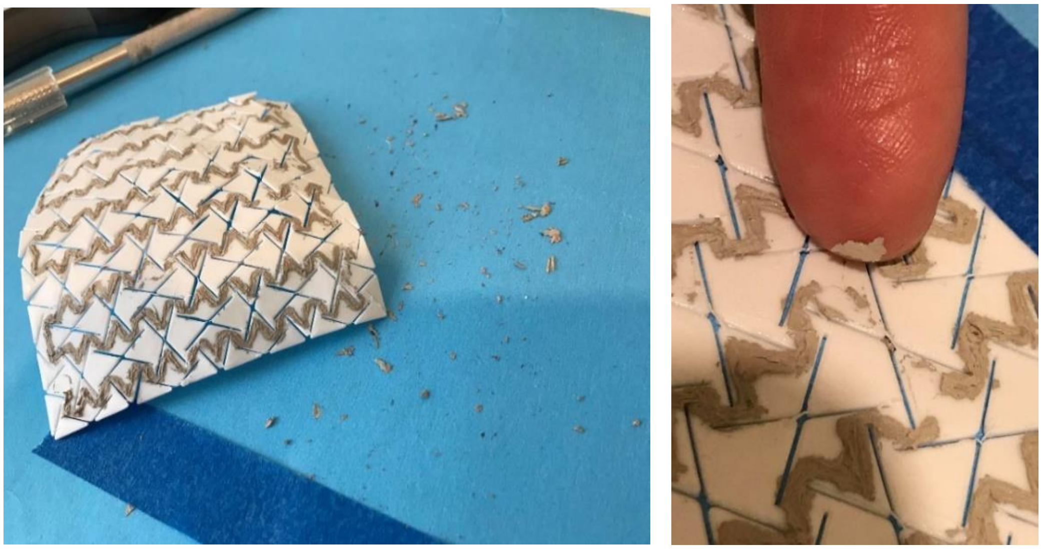


Fig. 3-17 Conductive ink was detached after bending the printed object

In conclusion after this exercise, several disadvantages were detected such as the material needs a process of curing, the feature of low-temperature curing of the ink provides suboptimal performance in terms of conductivity since conductivity performance depends on curing, and finally the detaching of the conductive ink that occurs after bending the geometry.

3.3.2 Printing the same geometry that can be bent but embedding conductive wire.

As a solution to the conductive ink detached, a new design was developed adding channels to the original geometry to embed a wire AWG30 as shown in **Figure 3-18.** The advantages of using wire are its high conductivity performance and durability.

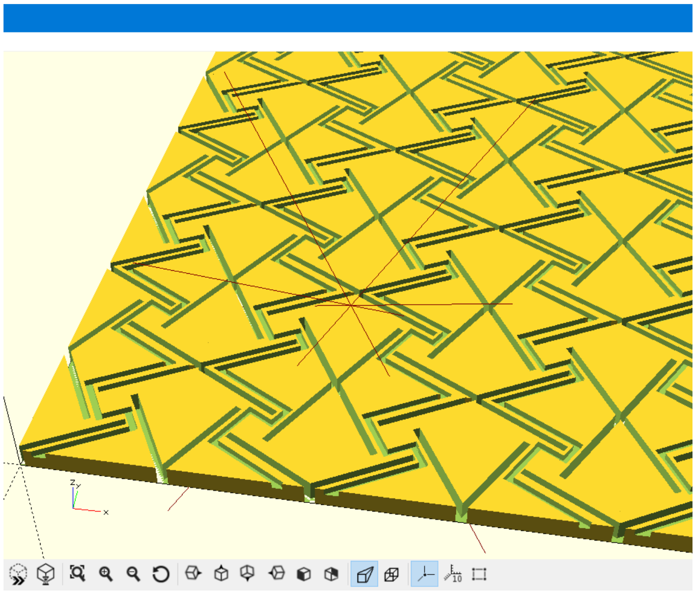


Fig. 3-18 Channels were added to the original geometry design for embedding a conductive wire.

To get enough resolution when 3D printing, a nozzle of 0.2mm was used to print the base which was $1.5mm$ in height and the channels were $0.5mm$ in depth as shown in **Figure 3-19**. After embedding the wire, a top of $0.5mm$ was printed to cover the wire embedded (**Figure 3-20**)

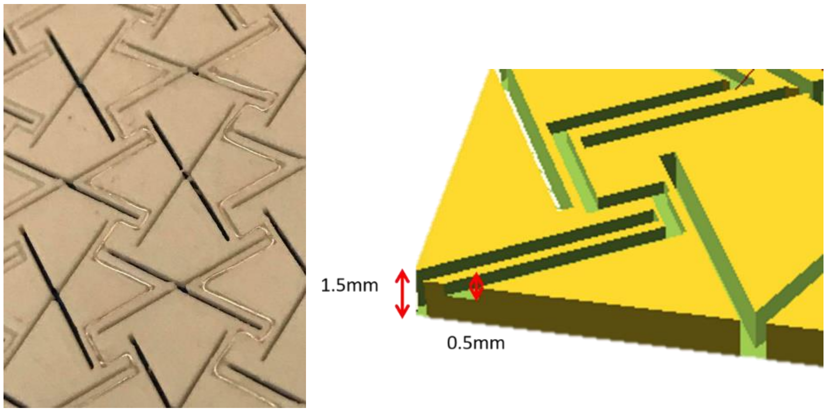


Fig. 3-19 Detail of the wire embedded into the channels and the depth of base and channels.

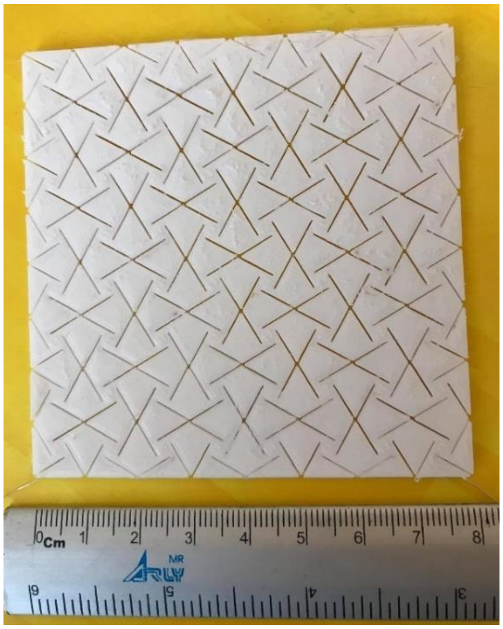


Fig. 3-20 The piece is finished with wire embedded.

To test any variation when the object is bent, first, the resistance of the wire was measured (0.6Ω) before and after bending but there was no significant variation, even measuring with a Wheatstone bridge. Then, a MyDAQ board from National Instruments was used to measure the frequency response of the sensor according to the connections in **Figure 3-21**.

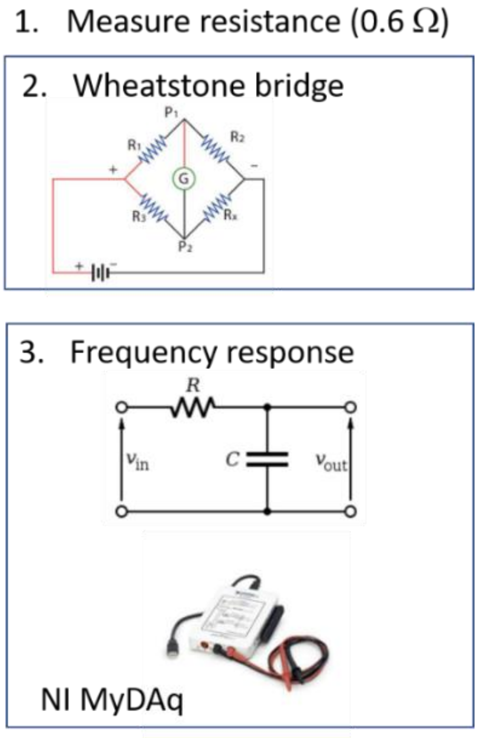
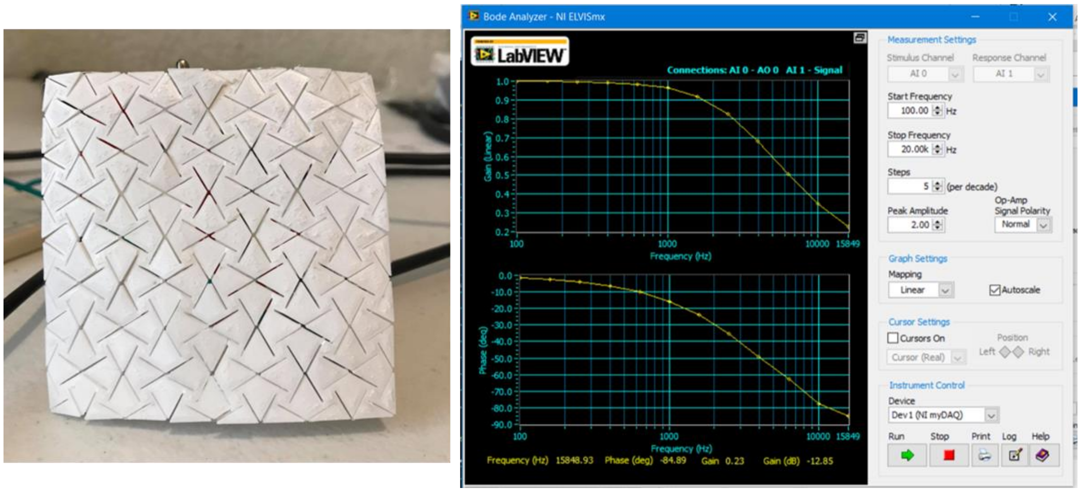


Fig. 3-21 The frequency response of the device was measured using a NI MyDAq board.

Three different states of the device were recorded: first when not folding happened, then when a partial folding, and finally when a significant bending. The results showed a variation in the phase of the frequency response from 10kHz. **Figure 3-22**, **Figure 3-23**, and **Figure 3-24** show how the gain was the same, but the phase suffered changes in a proportional way to the folding state of the piece.


Bode Analyzer - NI ELVISmx

Connections: AI 0 - AO 0 AI 1 - Signal

Frequency (Hz) 15848.93 Phase (deg) -84.89 Gain 0.23 Gain (dB) -12.85

Measurement Settings

Stimulus Channel: AI 0

Response Channel: AI 1

Start Frequency: 100.00 Hz

Stop Frequency: 20.00k Hz

Steps: 5 (per decade)

Peak Amplitude: 2.00

Op-Amp Signal Polarity: Normal

Graph Settings

Mapping: Linear

☑ Autoscale

Cursor Settings

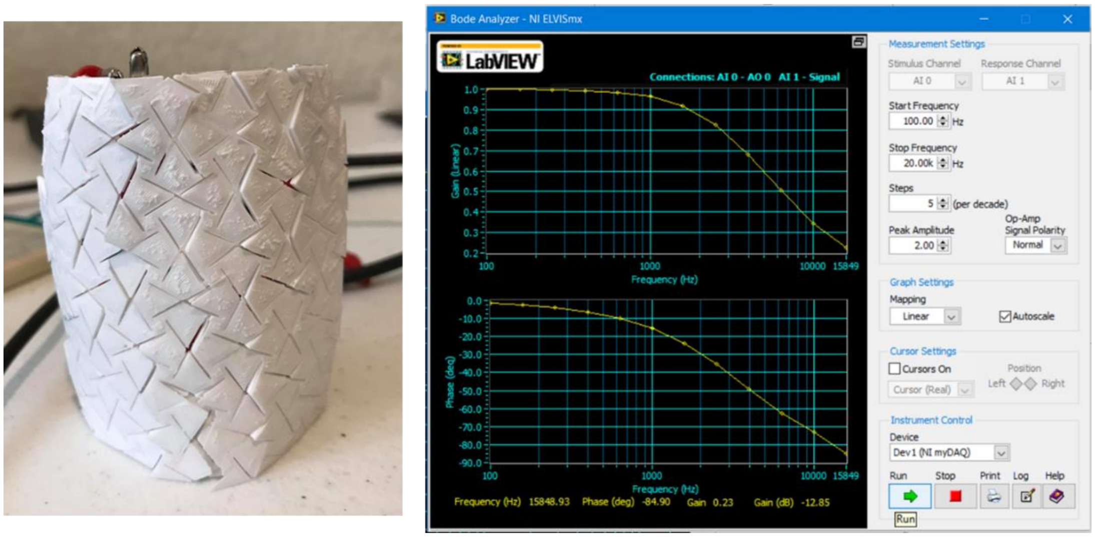
☐ Cursors On

Cursor (Real)

Instrument Control

Device: Dev 1 (NI myDAQ)

Fig. 3-22 When no folding took place, the phase was nearly -79 degrees at 10kHz.



Connections: AI 0 - AO 0 AI 1 - Signal

Measurement Settings
Stimulus Channel: AI 0
Response Channel: AI 1
Start Frequency: 100.00 Hz
Stop Frequency: 20.00k Hz
Steps: 5 (per decade)
Peak Amplitude: 2.00
Op-Amp Signal Polarity: Normal

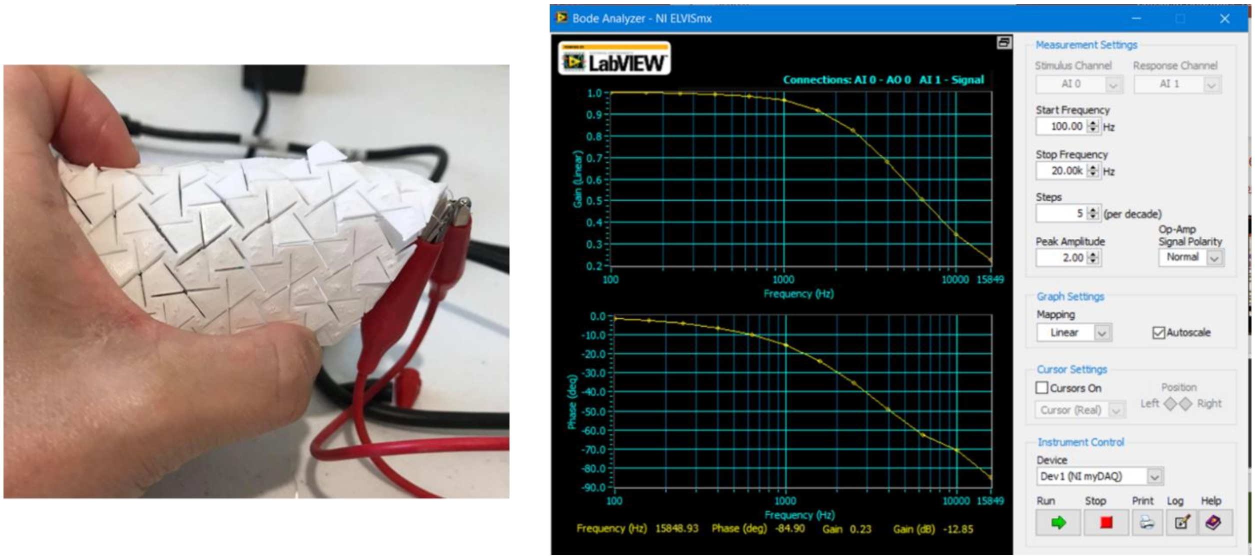
Graph Settings
Mapping: Linear
☑ Autoscale

Cursor Settings
☐ Cursors On
Cursor (Real)

Instrument Control
Device: Dev1 (NI myDAQ)

Frequency (Hz) 15848.93 Phase (deg) -84.90 Gain 0.23 Gain (dB) -12.85

Fig. 3-23 When partial folding took place, the phase was nearly -74 degrees at 10kHz.



Connections: AI 0 - AO 0 AI 1 - Signal

Measurement Settings
Stimulus Channel: AI 0
Response Channel: AI 1
Start Frequency: 100.00 Hz
Stop Frequency: 20.00k Hz
Steps: 5 (per decade)
Peak Amplitude: 2.00
Op-Amp Signal Polarity: Normal

Graph Settings
Mapping: Linear
☑ Autoscale

Cursor Settings
☐ Cursors On
Cursor (Real)

Instrument Control
Device: Dev1 (NI myDAQ)

Frequency (Hz) 15848.93 Phase (deg) -84.90 Gain 0.23 Gain (dB) -12.85

Fig. 3-24 When significant folding took place, the phase was nearly -70 degrees at 10kHz.

Embedding the wire showed promising results since the resistivity was almost zero. On the other hand, the deposition of the conductive silver ink requires a sophisticated procedure (application-curing-application-curing), and the properties of the conductive ink did not produce the expected behavior to work as a flex sensor.

3.3.3 Design and manufacturing of a multilayer printed spiral winding inductor.

To prove the applicability of embedding a wire into a functional device, the printing trajectories for a multilayer spiral winding were generated. Due to the limitations of the available equipment, the wire was embedded by hand pausing the 3D printing process. **Figure 3-25** shows a scheme of the strength of the magnetic field H $(3-9)$ generated when a current I (Amperes) flows through N turns of wire of length L (meters) forming a multilayer spiral coil as in **Figure 3-26**.

$$H = \frac{I \times N}{L} \qquad (3-9)$$

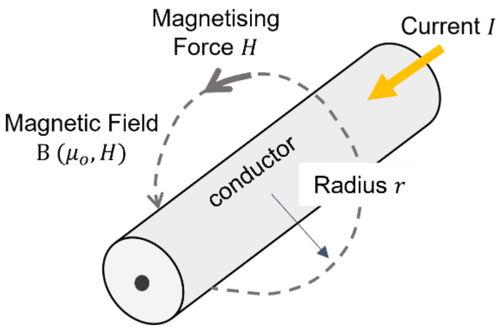


Fig. 3-25 Magnetic fields generated when passing a current through a wire.

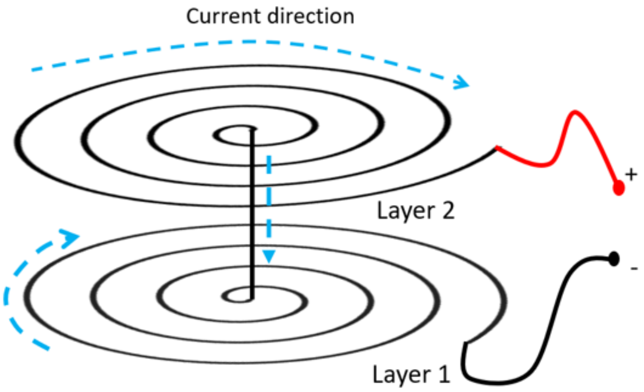


Fig. 3-26 Multilayers of coils to produce a magnetic field.

The behavior of the magnetic field for the designed coil was simulated obtaining the results in **Figure 3-27. Figure 3-28** shows the generation of the printing trajectories for each layer depending on if they were a non-conductive layer or the corresponding conductive layer with the channels to insert the wire.

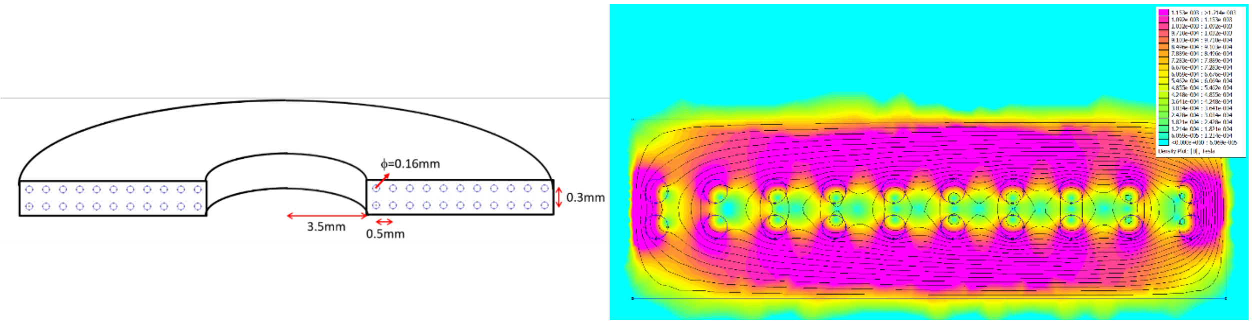


Fig. 3-27 Simulation of the magnetic field generated in the inductor with five layers alternated with conductive and non-conductive material.

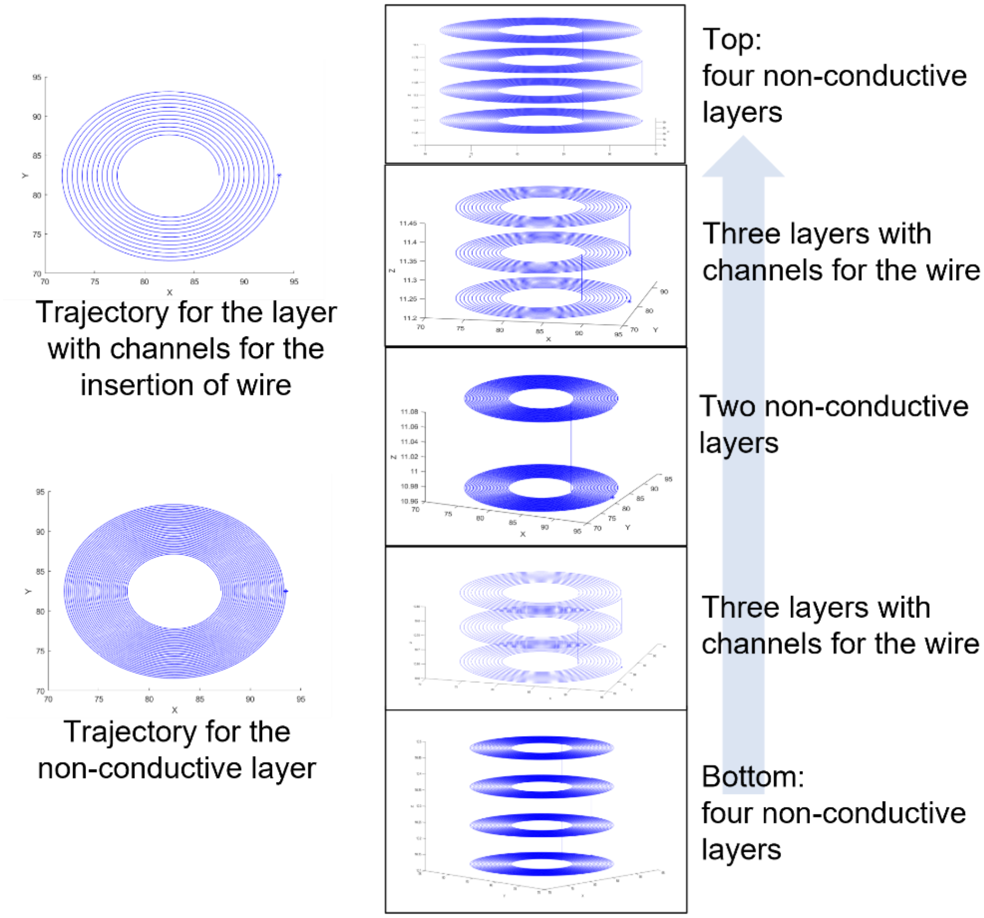
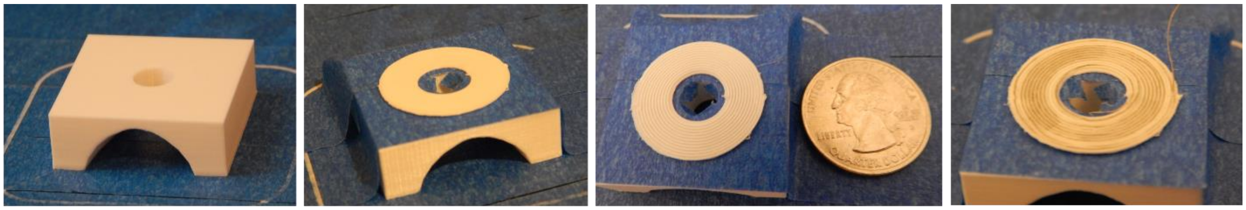


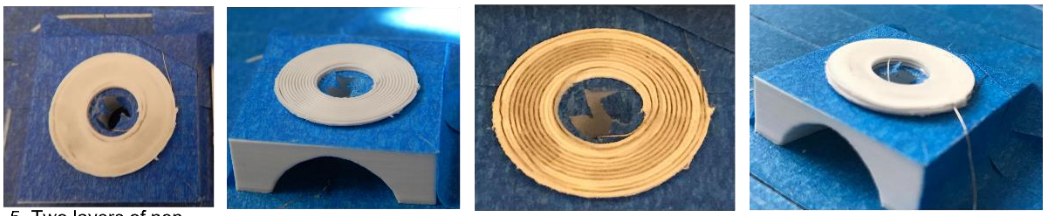
Fig. 3-28 Printing trajectories for the non-conductive and conductive layers of the inductor.

During the experimental implementation, the challenge was the proper hand-insertion of the wire to achieve very small distances between the turns of the wire to produce a proper enough magnetic field to be detected. **Figure 3-29** shows the steps followed for the manufacturing of the multilayer printed inductor which consisted of the following:

1) Printing of a base for easy insertion and manipulation of the wire between the layers. Covering this base with painter's tape.
2) Four layers of adjacent circular trajectories as the non-conductive section.
3) Three layers of spaced circular trajectories form channels for the insertion of the wire.
4) Insertion of the wire following the channels (conductive section).
5) Two layers of adjacent circular trajectories as the non-conductive section.
6) Three layers of spaced circular trajectories form channels for the insertion of the wire.
7) Insertion of the wire following the channels (conductive section).
8) Four layers of adjacent circular trajectories as the non-conductive section.



1. Base for proper manipulation of wire. 2. Four layers of non-conductive material (the bottom). 3. Three layers to form the channels for the insertion of wire. 4. Insertion of the wire.



5. Two layers of non-conductive material. 6. Three layers with channels for the insertion of wire. 7. Insertion of the wire. 8. Four layers of non-conductive material (the top).

Fig. 3-29 Procedure for the manufacturing of the inductor, alternating non-conductive and conductive material.

Using four rechargeable batteries of 3.7V/600mA serially connected, the printed inductor was tested by observing the presence of a magnetic field through a compass when the voltage was applied as in **Figure 3-30**.

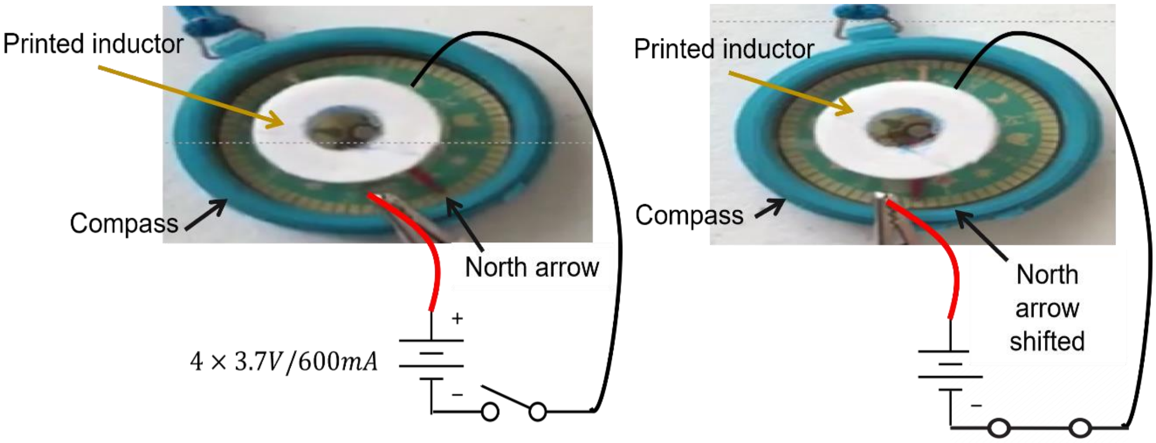


Fig. 3-30 Observing the presence of the magnetic field using a compass.

The possibility of printed inductors would allow planar and non-planar inductors inserted into a mechanical design, opening an endless number of applications such as robotics and mechatronics devices.

3.3.4 Touch sensor using conductive thread and filament TPU.

A pressure sensor using conductive thread and thermoplastic polyurethane (TPU) filament was proposed. TPU is a very flexible and elastic material, while the conductive thread is very sensitive to movement or pressure. The idea of the pressure sensor is that the conductive thread was inserted into the printed device through channels following a zig-zag path, and by pausing the printing process as shown in **Figure 3-31**.

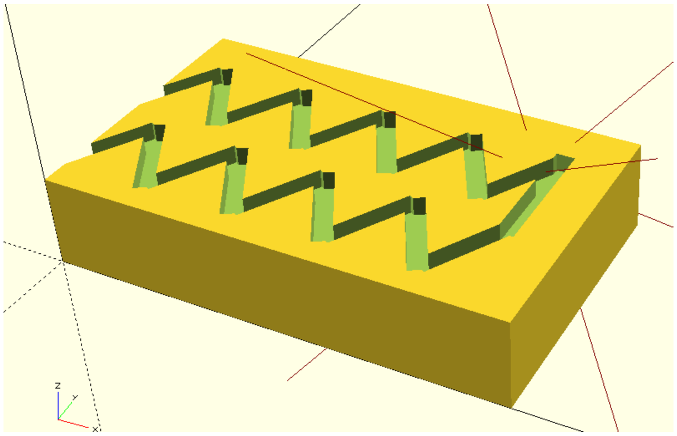


Fig. 3-31 Channels incorporated into the design of pressure sensor for embedding conductive thread.

Due to TPU material tending to clog the nozzle, it was used a nozzle of 0.4mm so, the width for channels was not so narrow as those used in section 3.8.2 because of the limitations in resolution. **Figure 3-32** shows the complete process followed to print the device using Cura for slicing the geometry, TPU filament, Ender-3 printer, nozzle of 0.4 mm, and conductive thread.

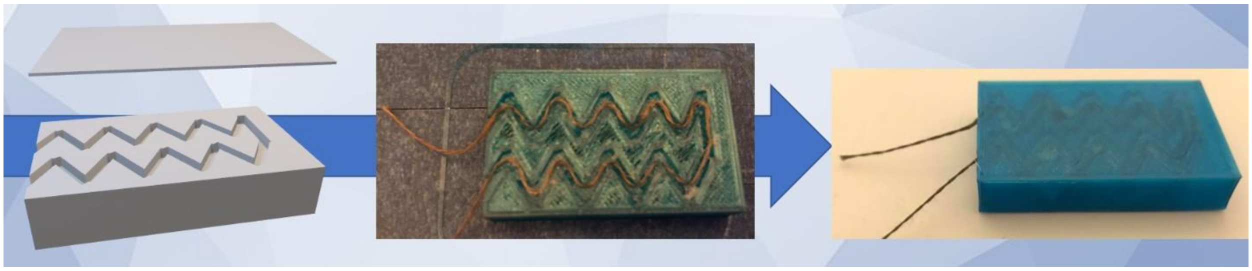


Fig. 3-32 Printing process paused to embed the conductive thread.

The performance of the sensor was very good even though the poor resolution of channels for the thread. The conductive thread is very sensitive to any movement, so the algorithm implemented for detection needs to distinguish between the false changes in voltage from those caused by pressing the sensor. Here, an Arduino board and a state machine were implemented to reject false changes in resistance.

Chapter 4 Trajectory planning for conformal printing on tessellated curved surfaces

In Chapter 2, it was mentioned that multi-axis tool path planning methods may be based on analytical or faceted geometries. In chapter 3, the printing trajectories were generated on analytical surfaces where the trajectories lying on a plane (e.g., XY) are translated into a new surface analytically defined. Having the new position of the printing trajectories, the machine code (G-code) may be generated and executed. On the other hand, the faceted or tessellated geometries (e.g., STL files) describe data using a very simple format having the advantage of portability between different-branded machines. The tessellated (STL) file format is the most used standard for the 3D printing industry.

This chapter is based on the publication of our paper (Rodriguez-Padilla et al., 2021), where was proposed an algorithm to achieve the deposition of material conformally on non-planar tessellated surfaces.

4.1 Algorithm to project a trajectory on a non-planar triangular mesh surface.

The intention of the algorithm proposed by (Rodriguez-Padilla et al., 2021) is to achieve the deposition of material conformally on non-planar tessellated surfaces following printing trajectories and the generation of the machine code for a 3D printer. The algorithm projects any sequential points, which could describe a trajectory or even any mesh of points, on a non-planar triangle-defined surface. The software used to implement the algorithm was Matlab2018. **Figure 4-1** shows the main idea of the algorithm implemented, where points are generated and projected on an arbitrary curved surface. **Figure 4-2** shows the flow diagram of the algorithm where each point, following a direction defined by a vector, crosses a triangle at a specific location. **Figure 4-3** shows some of the experimental results obtained where the step-effect is clearly observed and the deposition of the material conformally is successfully executed. One of the contributions of this work is a methodology to generate complex printing trajectories using an L-system and its applicability to lattice manufacturing.

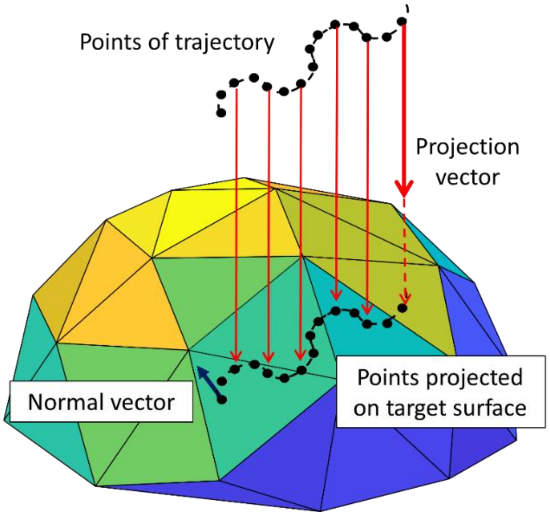


Fig. 4-1 Points in order are projected onto a curved tessellated surface.

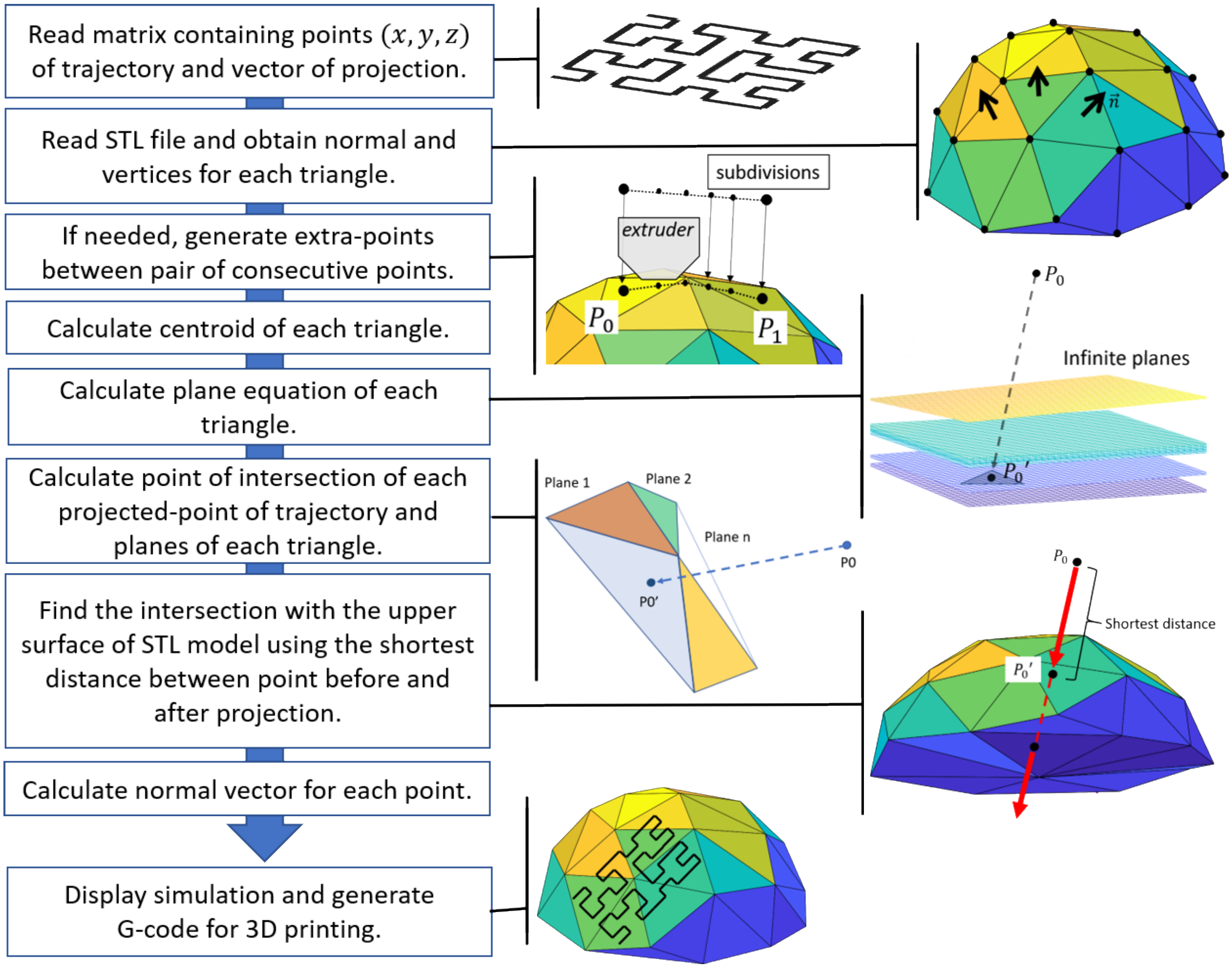

Read matrix containing points (x, y, z) of trajectory and vector of projection.
Read STL file and obtain normal and vertices for each triangle.
If needed, generate extra-points between pair of consecutive points.
Calculate centroid of each triangle.
Calculate plane equation of each triangle.
Calculate point of intersection of each projected-point of trajectory and planes of each triangle.
Find the intersection with the upper surface of STL model using the shortest distance between point before and after projection.
Calculate normal vector for each point.
Display simulation and generate G-code for 3D printing.
$\vec{n}$
subdivisions
extruder
P_0
P_1
P_0
Infinite planes
P_0'
Plane 1
Plane 2
Plane n
P0
P0'
P_0
Shortest distance
P_0'

Fig. 4-2 Flow diagram of the algorithm, where a point crosses a triangle.

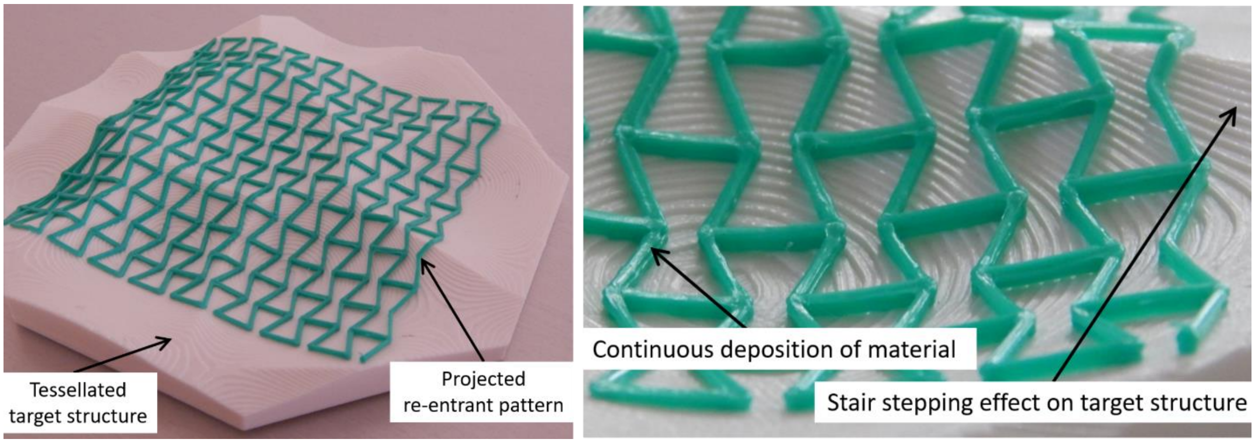

Tessellated target structure
Projected re-entrant pattern
Continuous deposition of material
Stair stepping effect on target structure

Fig. 4-3 Experimental results of a lattice projected on an arbitrary surface.

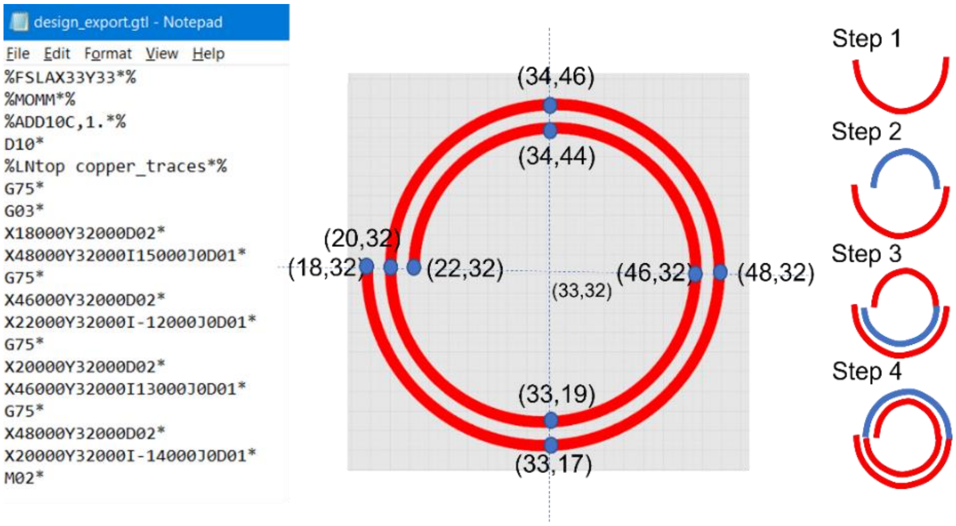
4.2 Generation of printing trajectories from a collection of data located inside a file.

The printing trajectories for lattices have been generated using iterative programming as explained in Chapter 3. However, it may be also generated from information formatted and found inside readable text files such as the Gerber files needed for the manufacturing of a PCB.

4.2.1 Extracting information inside a copper Gerber file.

Several Gerber files are produced after an electrical schematic has been designed and the PCB produced. Gerber files contain all the information necessary to manufacture a Printed Circuit Board such as the location of conductive tracks, pads, components, etc. for each layer. According to the Gerber file format specification, the Gerber file is an ASCII vector image file format representing 2D binary images. A Gerber file is a list of commands that creates graphics objects such as copper traces, vias, pads, and solder masks. Each image of each physical board layer of the PCB design needs one Gerber file, and these are human-readable. For a complete specification of the information contained inside the Gerber files, please refer to (Ucamco, 2014).

Here, the main interest is in the interpretation of the information contained in the Gerber file related to the conductive tracks of the copper layers of a PCB. As an example, refer to **Figure 4-4** in which the PCB contains just one layer and a conductive coil that was drawn on specific positions to be analyzed. The Gerber file associated with the copper layer contains the commands needed to define the shape of the tracks as a list of semicircles to form the complete coil. In this case, it can be observed that following the list of commands from top to bottom in the copper Gerber file, the trajectory generated is not a continuous trace.



```
%FSLAX33Y33*%
%MOMM*%
%ADD10C,1.*%
D10*
%LNtop copper_traces*%
G75*
G03*
X18000Y32000D02*
X48000Y32000I15000J0D01*
G75*
X46000Y32000D02*
X22000Y32000I-12000J0D01*
G75*
X20000Y32000D02*
X46000Y32000I13000J0D01*
G75*
X48000Y32000D02*
X20000Y32000I-14000J0D01*
M02*
```

Fig. 4-4 Commands inside the Gerber copper file (left) that produce the conductive coil track do not follow a continuous trajectory (right).

4.2.2 Algorithm to generate a trajectory from the information contained in a Gerber copper file (PCB).

The purpose of this algorithm is the generation of a continuous-as-possible trajectory from the commands contained in the copper Gerber file. As an example, the PCB in **Figure 4-5** was generated using online software https://upverter.com/. The circuit design corresponds to a simple thermometer using a sensor, comparators, LEDs, and resistors.

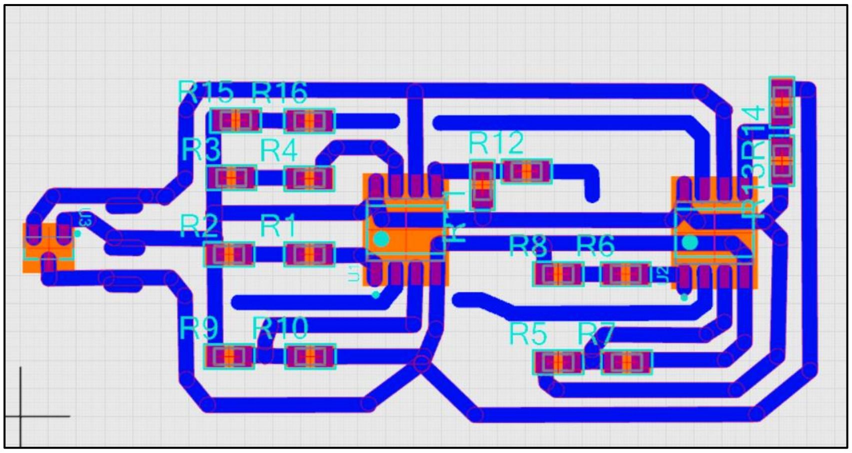
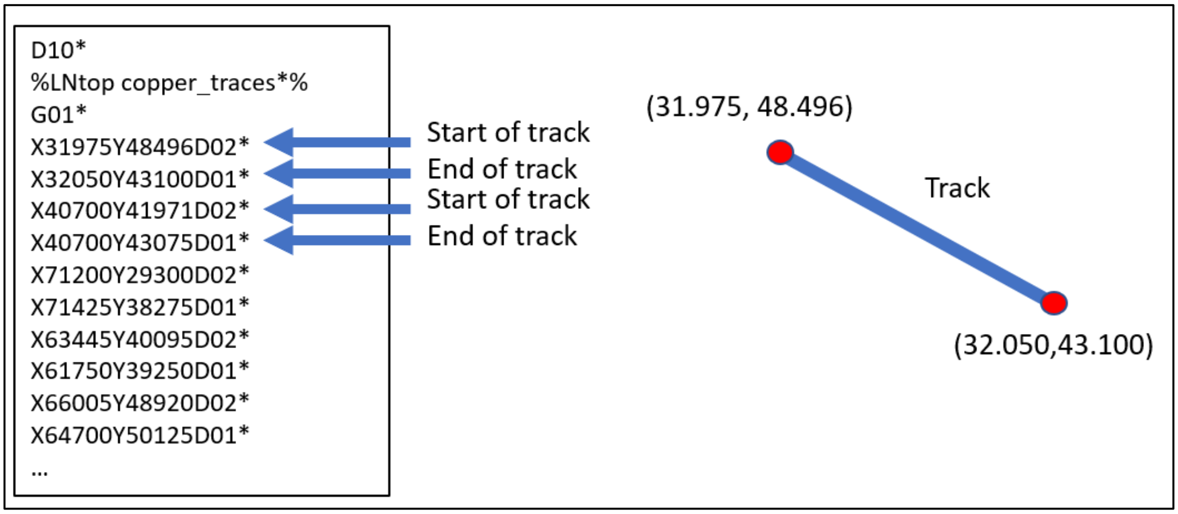


Fig. 4-5 PCB generated using upverter.com

Figure 4-6 contains part of the copper Gerber file information, where it can be observed the commands corresponding to the start and end of each conductive track.

This sequence of commands does not generate a continuous trajectory as explained before. For our purpose of 3D printing, the goal is the deposition of conductive material in a continuous-as-possible trajectory, looking for the smallest number of movements without extruding material. So, knowing that trajectory planning is very important for 3D printing, the algorithm developed measures the length of each track, sorts them, and finds the best trajectory for 3D printing avoiding discontinuous tracks that may cause interruption of the conductivity. The number of start/ends of the tracks may be very large and hence the searching time loop of the algorithm as well. To speed up the process to find the closest next start/end point, the PCB is divided into sections, the current start/end would look for the next point first inside its section which involves much fewer calculations. In case the end of a start/end of the track belongs to a different section, the searching is moved to the next section and so on until finishing with all the tracks. **Algorithm 4-1** contains the pseudocode and **Figure 4-7** shows the general idea of the implemented algorithm. Until now, the conductive tracks lie on an X-Y plane. Having the location of the start/end of each segment to be printed, parametric equations are used for the generation of the complete trajectory, and the G-code is to control a conventional 3D printer.



```
D10*
%LNtop copper_traces*%
G01*
X31975Y48496D02*
X32050Y43100D01*
X40700Y41971D02*
X40700Y43075D01*
X71200Y29300D02*
X71425Y38275D01*
X63445Y40095D02*
X61750Y39250D01*
X66005Y48920D02*
X64700Y50125D01*
...
```

Fig. 4-6 At the left, is the data contained in the copper Gerber file. On the right, it is the meaning of the two first commands.

Algorithm 4-1 Generation of optimized trajectory from Gerber copper file

---

Input: Copper Gerber file
Output: List of start/end of conductive tracks already sorted (optimized)

```
Open and read the copper Gerber file
Extract from the file the start/end of each conductive track
```

```
Calculate the length of each track and sort them
Choose the longest track as track1 (the beginning of the trajectory)
Divide the PCB area into 3x3 sections
for all_tracks do
    for all_sections do
        Look for the nearest start of the track2 within the same section
        Jump to the next section in case of start of the track2 is not within the current section
    Store track1
    track1 ← track2
```

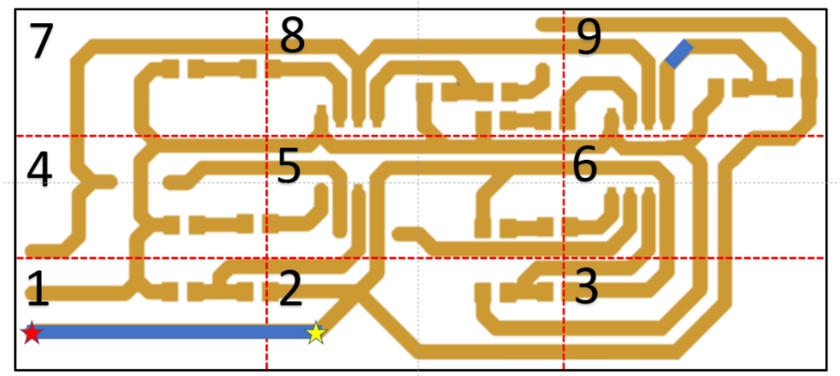


Fig. 4-7 Suggested algorithm which divides the PCB into 3×3 sections to speed up the searching loop to find an optimal path printing.

Once the data is extracted and the algorithm finds the best sequence to follow the conductive tracks optimizing the movements of the extruder with the filament contracted, the start/end of each segment can be represented using parametric equations for the generation of the G-code for the printer.

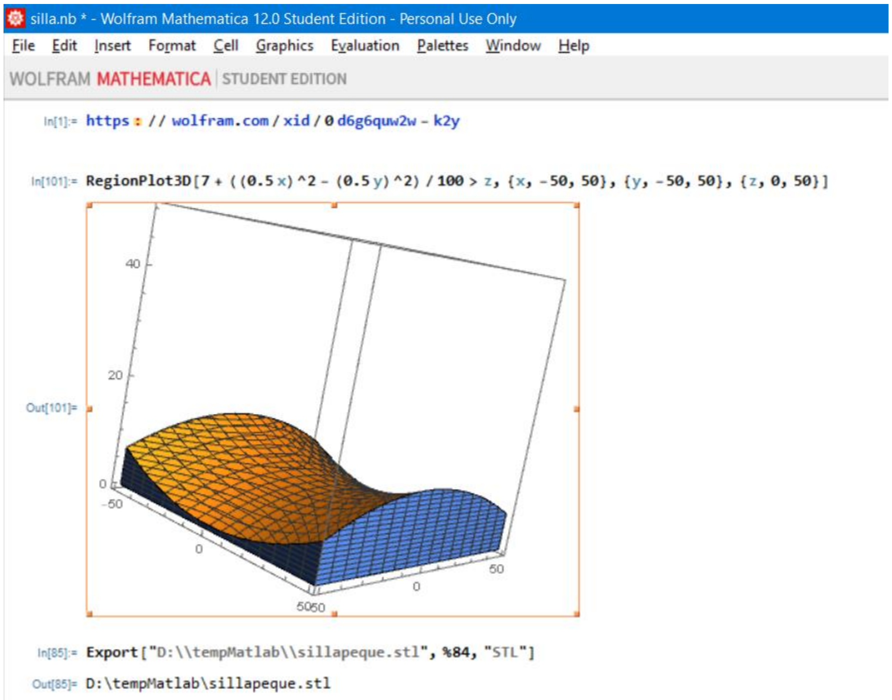
4.2.3 Printing trajectories from the Gerber copper file of a circuit.

Using the algorithm explained in section 4.1 (Rodriguez-Padilla et al., 2021), the conductive tracks extracted from the information inside the Gerber copper file can be used to generate parametric trajectories and then project them on any non-planar tessellated surface.

As a first example, a saddle surface was generated analytically using $(4-1)$ and the software Mathematica. The geometry was exported as a tessellated surface (STL file) as in **Figure 4-8**

$$z = 7 + \frac{(0.5x)^2 - (0.5y)^2}{100} \qquad (4-1)$$

The STL file generated was used as an input for the algorithm to project the trajectories of the PCB as shown in **Figure 4-9,** and to generate the G-code for the printing trajectories of the PCB. **Figure 4-10** shows the simulation where the new location of the points and their normal vector can be visualized (**Figure 4-11**)


silla.nb * - Wolfram Mathematica 12.0 Student Edition - Personal Use Only
File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
WOLFRAM MATHEMATICA STUDENT EDITION
40
20
0
-50
0
5050
0
50

```
In[1]:= https : // wolfram.com / xid / 0 d6g6quw2w - k2y

In[101]:= RegionPlot3D[7 + ((0.5 x)^2 - (0.5 y)^2) / 100 > z, {x, -50, 50}, {y, -50, 50}, {z, 0, 50}]

Out[101]=

In[85]:= Export["D:\\tempMatlab\\sillapeque.stl", %84, "STL"]

Out[85]= D:\tempMatlab\sillapeque.stl
```

Fig. 4-8 Mathematica software used to generate the geometry corresponding to a saddle surface according to $(4-1)$.

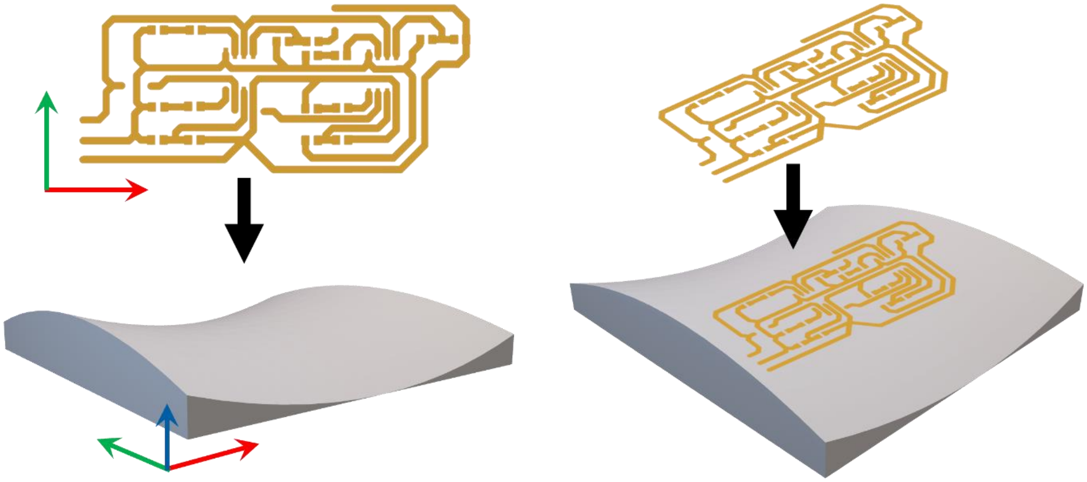


Fig. 4-9 Conductive tracks extracted from the Gerber file and projected on tessellated saddle geometry.

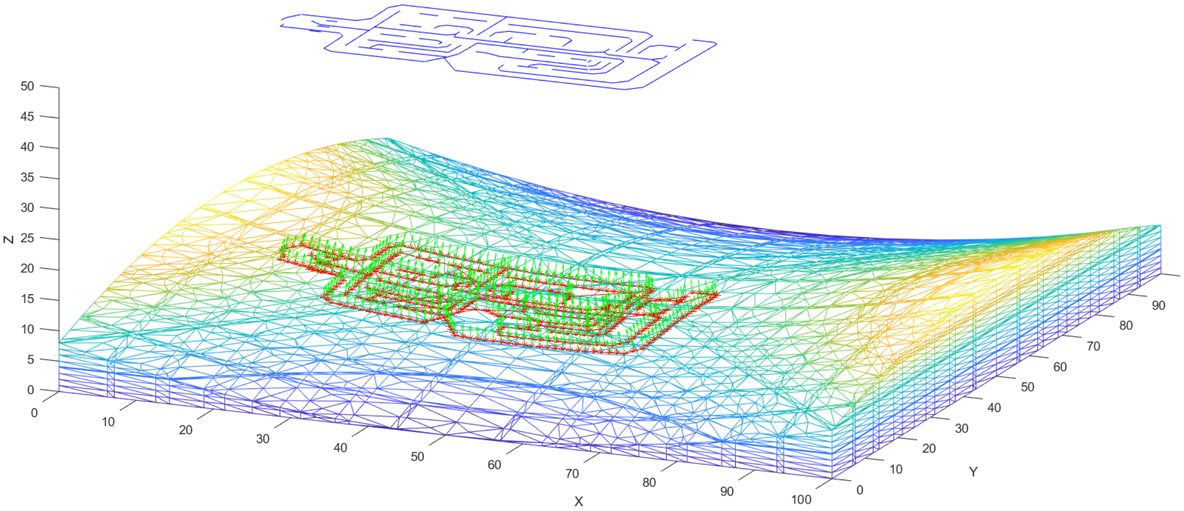


Fig. 4-10 Simulation of the trajectories of conductive tracks extracted from a Gerber copper file and then projected on a saddle surface.

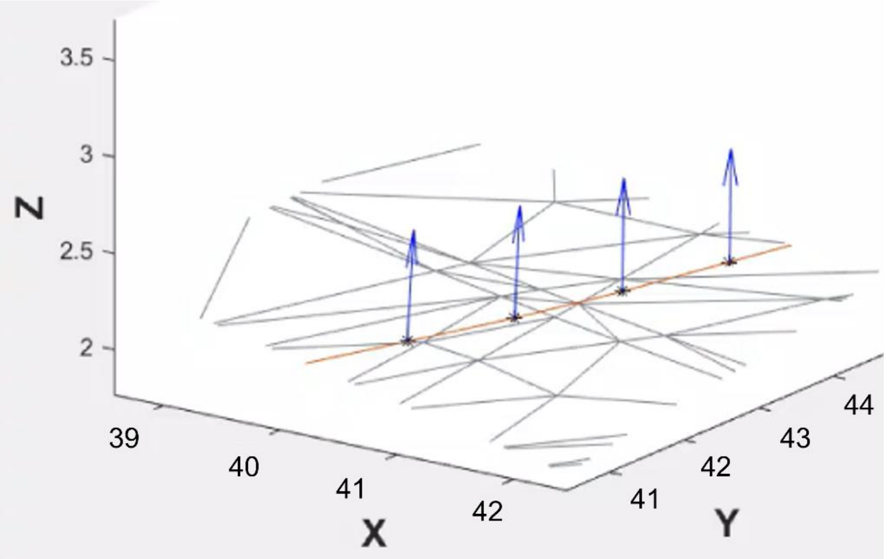


Fig. 4-11 View in detail where each projected point has its surface normal vector.

The results obtained prove that the algorithm proposed to project points as a printing trajectory may be used as well to generate printing trajectories from data contained inside readable files as one of the Gerber files that contain the conductive traces information.

4.3 Lindenmayer system (L-system) for the methodological generation of printing patterns.

The generation of lattices that are repetitive patterns may be achieved using Lindenmayer systems. An L-system consists of symbols used to make strings according to some production rules. The L-system starts with an axiom, and each symbol represents an action. Geometric structures can be generated in a recursive way to form fractals or model the growth process of the development of a plant or cell (Aref, 1991). Here, the L-systems are used to generate different patterns found in lattices.

As an example, consider **Figure 4-12a**, in which a Hilbert pattern has an initiation seed equal to “X”, where "X" = " $-YF+XFX+FY-$ ", “Y" = " $+XF-YFY-FX+$ ", “ $-$ “ means *turn right* $\pi/2$ *degrees*, “ $+$ ” *means turn left* $\pi/2$ *degrees* and “F” means *go Forward* A *distance*. Similarly, for the re-entrant curve (**Figure 4-12b**), the seed is “X” where "X" = “ $+\mathrm{F}-\mathrm{G}-\mathrm{F}+\mathrm{G}\,\mathrm{X}$”, “G” means *go Forward* B *distance* repeated *N* times to form a row and then *M* times to form the lattice. In this case, the angle of rotation is $2\pi/3$ and two different distances, F and G, are considered. A hexagonal pattern is a re-entrant pattern with distance $\mathrm{A}=\mathrm{B}$, and a turning angle of $\pi/3$ for the turning actions.

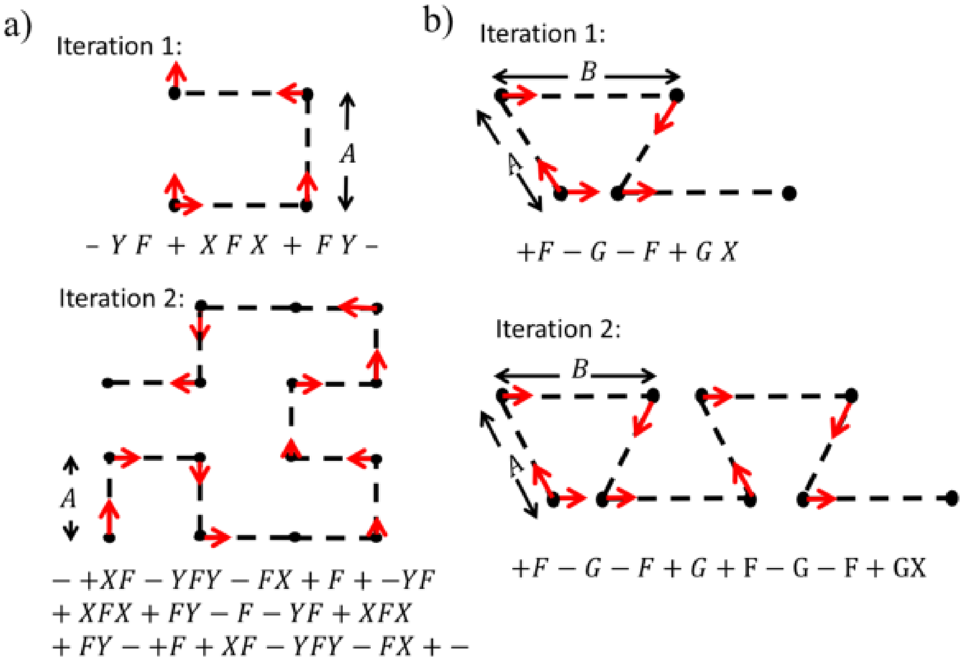

a)
Iteration 1:
A
- Y F + X F X + F Y -
Iteration 2:
A
− +XF − YFY − FX + F + −YF
+ XFX + FY − F − YF + XFX
+ FY − +F + XF − YFY − FX + −
b)
Iteration 1:
B
A
+F − G − F + G X
Iteration 2:
B
A
+F − G − F + G + F − G − F + GX

Fig. 4-12 First and second iterations for a) Hilbert pattern and b) re-entrant pattern using L systems.

4.4 Conclusions

Three procedures to generate trajectories have been explained: 1) using parametric equations having control of the start and end of the trajectory through the parameters (u, v, w), 2) using L-systems where a seed is generated and by a recursive iteration the seed starts growing generating repetitive patterns as the re-entrant, the hexagonal, and the Hilbert patterns, 3) from data contained in a readable file as the Gerber conductive file where a sorting and optimization process was proposed to reduce the number of retractions of the filament while the deposition of the material.

Chapter 5 Multi-axis system for 3D printing

In Chapters 3 and 4, the limitations of using a system of 2.5 degrees of freedom for the achievement of curved printing were exposed needed as a solution for the implementation of a multi-axis system. Here, a manipulator's arm is used to complete the layout for freeform manufacturing design and a requirement for the curved printing trajectories. Chapter 5 contains the two more important aspects of the proposed design: the hardware and the software implementation.

5.1. Complete hardware system for 3D printing

Figure 5-1 contains a general diagram of the system implemented to achieve multi-axis 3D printing using robotics. The main elements are the robot UR3 (Universal Robots), the whole extruder (Flexion) (https://flexionextruder.com/) attached to the robot as a tool, a microcontroller system based on the RAMPS 1.6 board inserted onto an Arduino Mega board, the build platform, and a Dell computer running MATLAB 2021 as the main control.

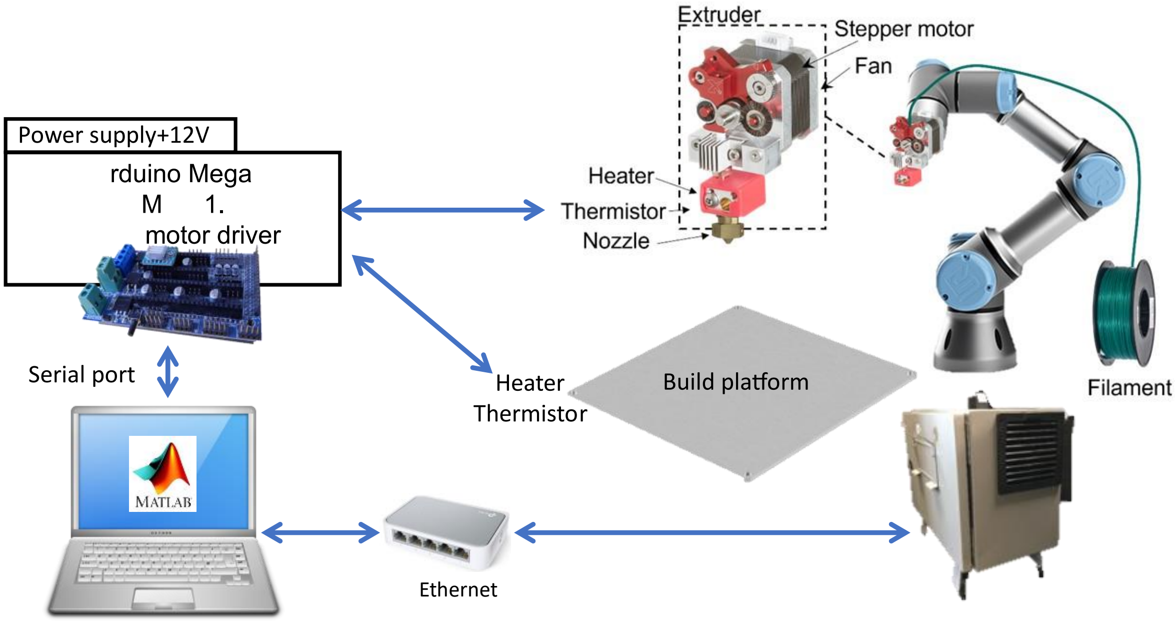


Fig. 5-1 System implemented for robotic curved conformal 3D printing.

5.1.1. The complete filament extruder mechanism.

Figure 5-2 shows how the extruder is attached to the manipulator's arm as a tool. The main parts of the extruder mechanism are: 1) Diabase Flexion extruder, 2) +12V fan, 3) NEMA17 stepper motor, 4) +12V heater element (ceramic), 5) nozzle and 6) temperature sensor (10kΩ-thermistor)

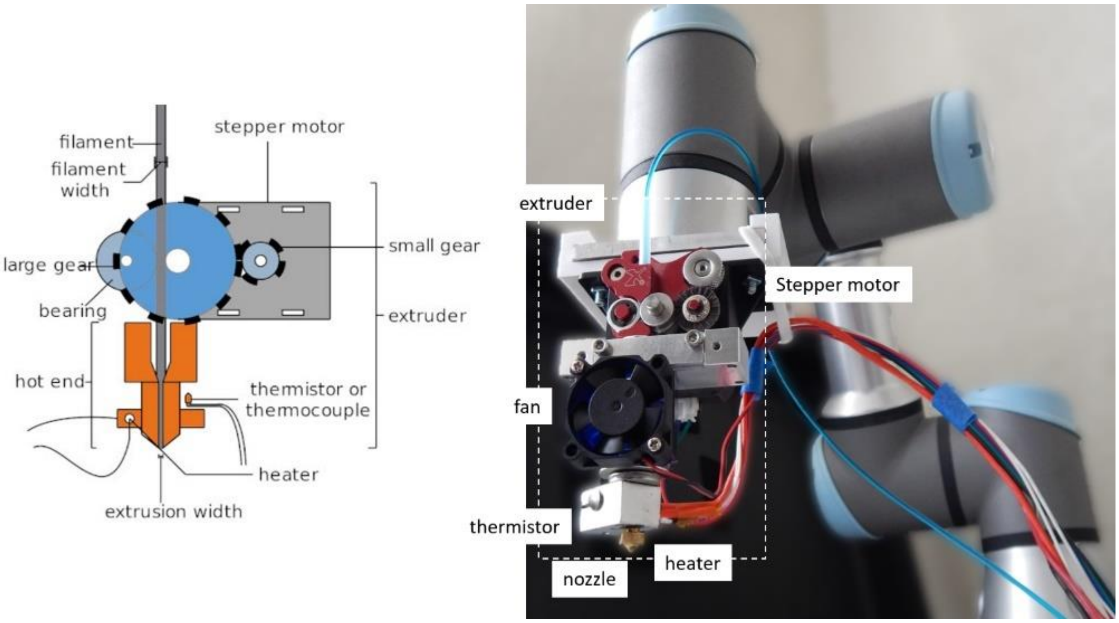


Fig. 5-2 Extruder attached as a tool to the manipulator's arm

5.1.2. Arduino-base microcontroller system

A Mega-Arduino board is used for the control of the different outputs and data acquisition from the sensors through a RAMPS 1.6 conditioner board. The behavior of the Arduino board is as a slave of the master located on a Dell computer running Windows 10 and MATLAB 2021. **Figure 5-3** shows the different inputs/outputs that the Arduino-based system has to control being its main functions:

- Read the temperature sensors (thermistors at the extruder and the build platform.)
- Control the temperature of the extruder and build a platform using a PID algorithm.
- Applied the proper Pulse Width Modulation (PWM) signals to each heater (extruder and build platform.)
- Control the power on/off for the fan at the extruder.
- Generate the pulses to move the stepper motor to extrude the filament.
- Establish serial communications to receive/transmit data from the master.

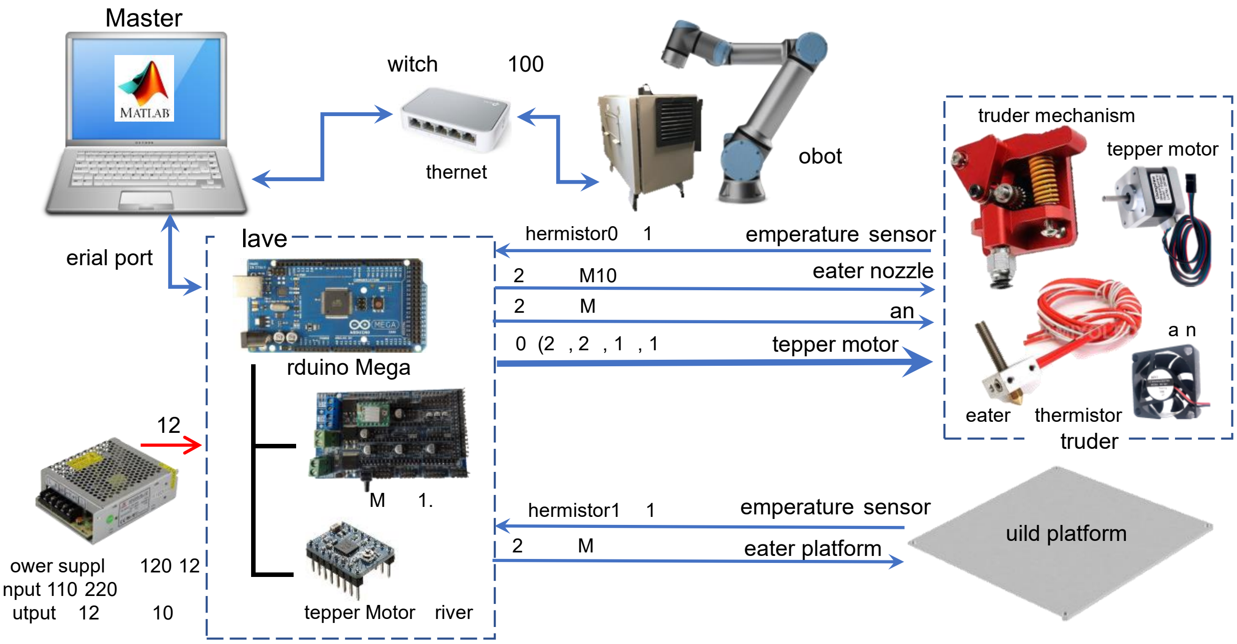


Fig. 5-3 Signals to the Arduino-based board to control the task of printing.

5.1.2.1. Ramps as a driver shield

RepRap (reprap.org/) Arduino Mega Pololu Shield (RAMPS) 1.6 is an open-source controller board used in do-it-yourself 3D printers and designed to be inserted on a Mega Arduino board. RAMPS board allows conditioning for the most important signals needed for the building of an open-source 3D printer such as the temperature sensors, the control signals, and the voltage driver for the stepper motors. Appendix D includes the connection between the Arduino Mega and the Ramps board, as well as the input/output of the signals.

5.1.2.2. Thermistor as a temperature sensor

To read temperatures for an extruder which may heat up to $240^{o}C$, a low-cost, very sensitive, and durable sensor is used such as the 10k-thermistor whose resistance decreases as the temperature increases. **Figure 5-4** shows the connections on the Ramps board with both thermistors.

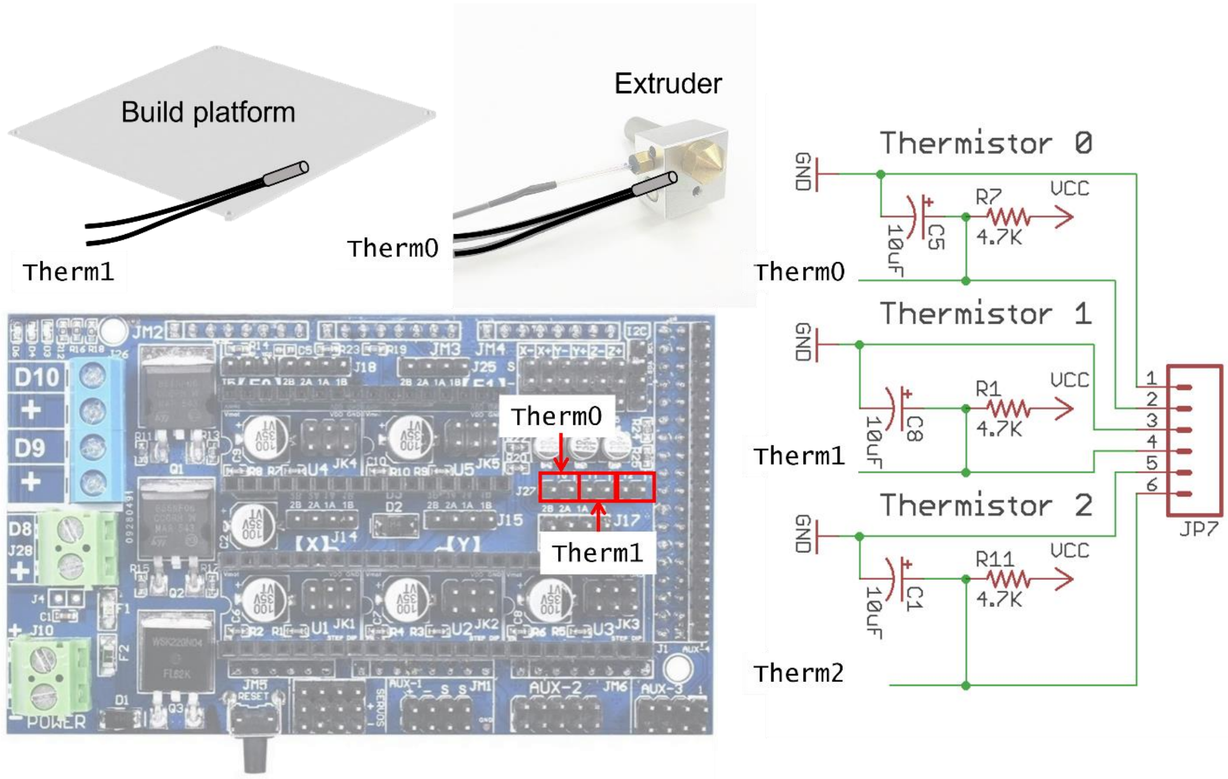


Fig. 5-4 Connection between the thermistors (located at the extruder and the build platform) and the RAMPS board.

Due to the non-linear response of the thermistors, a characterization function was obtained after a comparison between the divisor voltage of the thermistor and an infrared thermometer. **Figure 5-5** shows the characterization curve obtained for both 10kΩ-thermistors and the electrical configuration. The transformation between voltage and temperature was done using these acquired values and adjusted linearly between the different missing values.

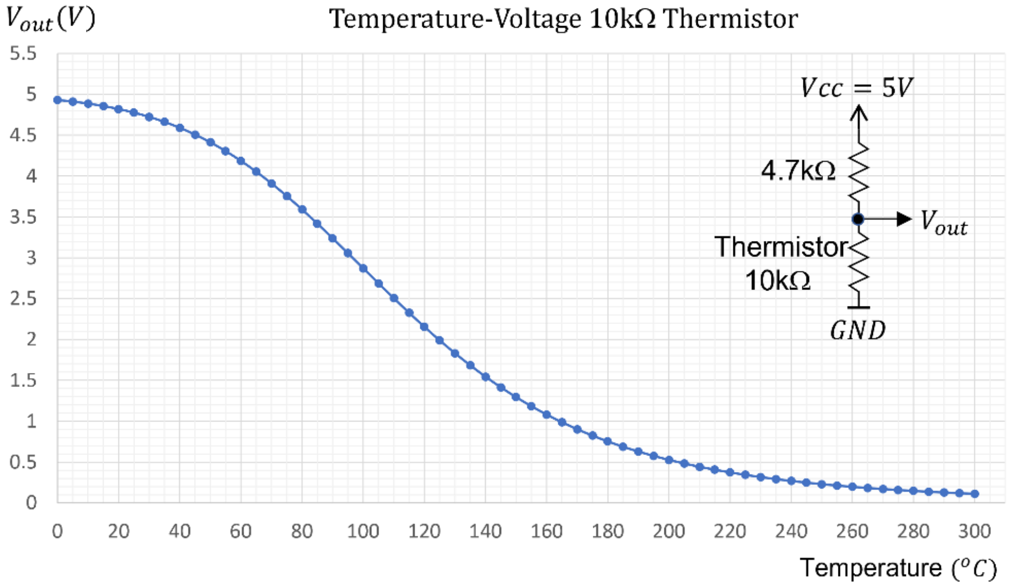


Fig. 5-5 Characterization of the thermistor using an infrared thermometer as reference

5.1.2.3 Ceramic heater

Both, the extruder and build platform, are heated up to the desired temperature through a ceramic cartridge heater. Typical temperatures for the build platform are 50-60°C, while the extruder reaches temperatures up to 240°C depending on the requirements of the filament material. **Figure 5-6** shows the electrical configuration (RAMPS board) to control the on/off of the heater.

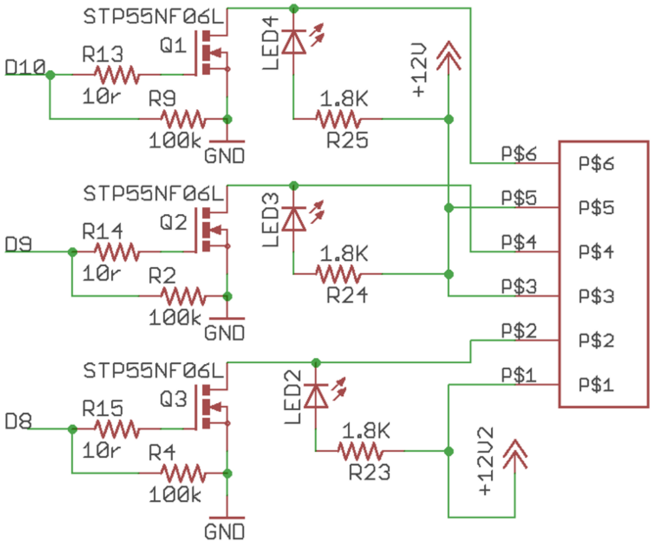
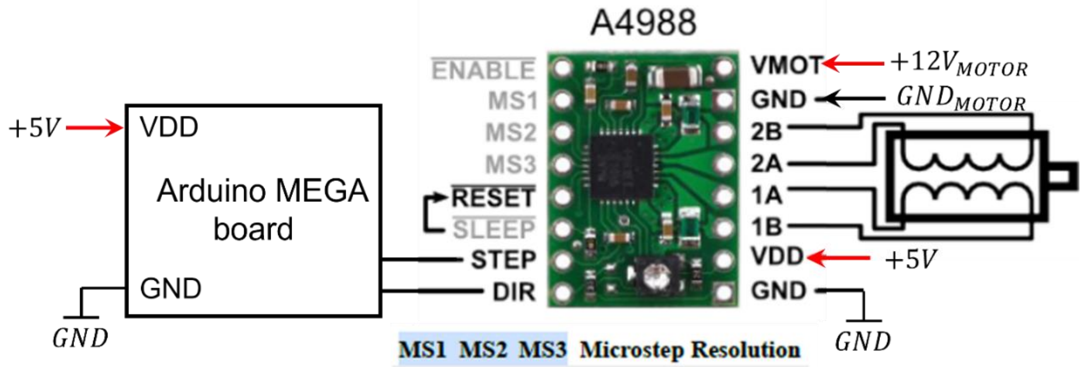


Fig. 5-6 Electrical configurations to control the heater

5.1.2.4 Extrusion of material through the stepper motor

The deposition of the filament is through the movement of a NEMA17 stepper motor. The steeper motor driver A4988 is used to control the stepper motor which has a resolution of 1.8^o (200 steps per revolution). The quantity and speed of extrusion are achieved by varying the frequency of the STEP signal and the direction of rotation with the DIR signal. After analyzing the performance of the extruder and stepper motor together, it was established that 300 steps in the motor extrude 25 mm of filament which means that the smallest quantity of filament to be deposited is 0.083mm without considering the funnel formed at the nozzle of 0.4mm, e.g., the filament is 1.75mm but the nozzle is 0.4mm. A typical printing speed in G-code, for instance, of 1500mm/min (25mm/s) would require 300 steps for the motor at a frequency of 300Hz to extrude 25mm of filament. **Figure 5-7** shows the connections between the motor driver A4988 and the microcontroller as well as the jumpers for the resolution of the micro-step. In this case, the sixteenth step was selected to have a maximum resolution.



MS1	MS2	MS3	Microstep Resolution
Low	Low	Low	Full step
High	Low	Low	Half step
Low	High	Low	Quarter step
High	High	Low	Eighth step
High	High	High	Sixteenth step

Fig. 5-7 Motor driver for NEMA-17 for the extrusion of filament.

5.1.2.5 Programming the Arduino

The programming of the Arduino Mega board includes two main tasks: 1) the reception of commands from the master and their execution, and 2) control of the target temperature (defined by the user) for the nozzle and the build platform.

5.1.2.5.1 Protocol of communication between the master computer and the slave Arduino-base board

The Arduino-base board aims at the function of the slave which executed the commands of the master, in this case, the master is the script running in Windows 10 and MATLAB 2021 on a Dell 5570 computer. The slave is responsible for data acquisition and control of the power of the heaters and the fan through PWM signals. The protocol implemented to achieve the handshaking of information between the slave and the master through the serial port is contained in **Table 5.1**

Command from master to slave	Response from slave to master	Description of command
"A"	*"A"*	Recognition of successful serial communication
"B"	*Temperature of nozzle*	Reading of current temperature of the nozzle
"C"	*Temperature of platform*	Reading the current temperature of the build platform
"D"+PWM	*"D"*	Setting the PWM value for the heater of the nozzle
"E"+PWM	*"E"*	Setting PWM value for heater of the build platform
"F" + NSteps	*"F"*	Turn motor N steps clockwise (extrusion)
"G" + NSteps	*"G"*	Turn motor N steps counterclockwise (retraction)
"H"	*"H"*	Turn fan on
"I"	*"I"*	Turn fan off
"L"	--	Read both temperatures of the thermistor (nozzle and build platform)
"M"	---	Write both PWM for the heater (nozzle and build platform)
"N"	---	Receive nozzle and platform temperature target
"O"	---	Stop the process of controlling temperature (write 0 to both heaters)

Table 5-1 Commands implements for data transfer between the master and slave.

5.1.2.5.2 Temperature control using a PID controller

The temperature control of the nozzle and the build platform was implemented using a Proportional Integrative Derivative (PID) controller through the ceramic heater and the thermistor, where the gain constants after tuning were fixed as follows: $k_{P_{nozzle}} = 60$, $k_{I_{nozzle}} = 100, k_{D_{nozzle}} = 20, k_{P_{platform}} = 10,\ k_{I_{platform}} = 20,\ k_{D_{platform}} = 10.$ **Algorithm 1** contains the pseudo-code for the implemented PID controller.

Algorithm 1 Temperature PID controller

```
Average five consecutive readings of sensor
Convert from voltage to degrees
$error \leftarrow targetTemp - currentTemp$
$lastError \leftarrow currentError$
$currentError \leftarrow error$
$pPID \leftarrow kp * currentError$
$iPID \leftarrow 0.01 * iPID + (ki * currentError)$
$dPID \leftarrow kd * (currentError - lastError)$
$PID \leftarrow pPID + iPID + dPID$
if $PID \leq 0$ then
    $PID \leftarrow 0$
else
    if $PID \geq 120$ then
        $PID \leftarrow 120$
    end if
end if
Write the PID as the PWM value
```

5.1.3 UR3/CB3 Robot

The requirements of a manipulator's arm for 3D printing, not printing on large scale, suggest a collaborative robot with at least 5 degrees of freedom, a load capacity of less than 1 kg, and a working space enough for printing on a table desk. The UR3 CB3 manufactured by Universal Robots (www.universal-robots.com) is a 6-axis robot arm whose specifications include allowing collaborative work, a working radius of 500mm, a maximum payload of 3kg, capabilities of communication with other equipment, analog, and digital inputs/outputs, and software named Polyscope to program the robot. As defined in the ser’ s Manual of the / by Universal Robots, **Figure 5-8** shows the dimensions of the robot and its configuration.

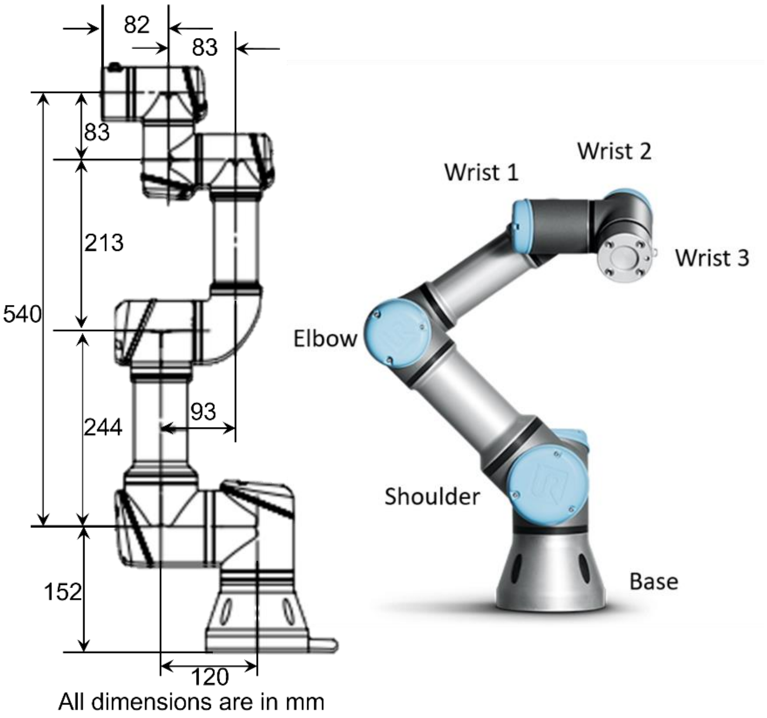


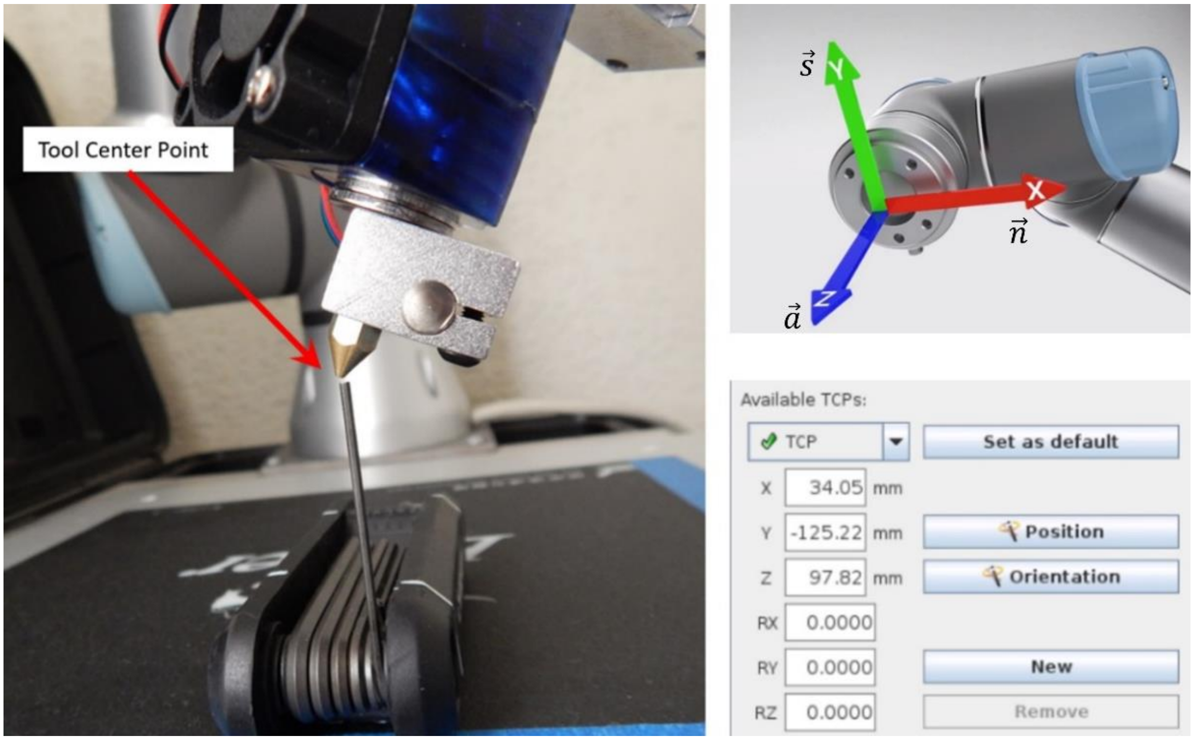
Fig. 5-8 Dimensions and configuration of arm robot UR3/CB3

5.1.3.1 Setting up the tool

A tool can be attached to the end of the arm (flange) of the robot, in this case, it consists of an extruder of filament. The Tool Center Point (TCP) is the part of the tool (end of the nozzle) to be in contact with the workpiece (build platform). The tool needs to be configured to set the following information:

- Location of TCP $(x_{tool}, y_{tool}, z_{tool}, Rx_{tool}, Ry_{tool}, Rz_{tool})$
- Weight of TCP (Extruder = 0.45 kg)
- Center of gravity (cog)

The UR3 uses the four-point method for the configuration of the TCP. This method consists of locating the TCP on the same point using four different orientations and then calculating the parameters of location $(x_{tool}, y_{tool}, z_{tool})$, and orientation $(Rx_{tool}, Ry_{tool}, Rz_{tool})$. **Figure 5-9** shows an example of how the extruder attached to the robot is configured as its TCP and the resulting position and orientation of the tool shown in the teach pendant.



Available TCPs:

TCP

X 34.05 mm

Y -125.22 mm

Z 97.82 mm

RX 0.0000

RY 0.0000

RZ 0.0000

Fig. 5-9 Configuration of the extruder as TCP of the robot

The homogenous matrix for the TCP (i.e., the extruder) is needed for its transformation with respect to the flange of the robot (frame {6}). In this case, the homogenous matrix is given by $(5-1)$ in millimeters, where the vectors $\vec{n}, \vec{s}, \vec{a}$ for the tool to have the same orientation that those of the flange of the robot as defined in **Figure 5-9**.

$$^{6}A_{extruder} = \begin{bmatrix} n_x & s_x & a_x & x_{tool} \\ n_y & s_y & a_y & y_{tool} \\ n_z & s_z & a_z & z_{tool} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 34.05 \\ 0 & 1 & 0 & -125.22 \\ 0 & 0 & 1 & 97.82 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5-1)$$

After the configuration of the TCP, the corresponding UR3 script instruction would be given by $set\_tcp(pose)$. According to the script Manual of Universal Robot, a *pose* is given as $p[x, y, z, \theta u_{x_{tool}}, \theta u_{y_{tool}}, \theta u_{z_{tool}}]$, where (x, y, z) is the position of the TCP given in meters, and $(\theta u_{x_{tool}}, \theta u_{y_{tool}}, \theta u_{z_{tool}})$ is the orientation of the TCP, given in *axis-angle* notation and radians. In this example, the instruction would be

$$set\_tcp(p[0.03405, -0.12522, 0.09782, 0.0, 0.0, 0.0])$$

The relationship between the *axis-angle* notation $(\theta u_{x_{tool}}, \theta u_{y_{tool}}, \theta u_{z_{tool}})$ and $roll\ (\gamma), pitch(\beta)$, and $yaw(\alpha)$ is given by equations $(5-2)$ through $(5-15)$

$$rv = \begin{bmatrix} \theta u_x \\ \theta u_y \\ \theta u_z \end{bmatrix} \qquad (5-2)$$

$$u_x = \frac{r_{32} - r_{23}}{2 * \sin\theta} \qquad (5-3)$$

$$u_y = \frac{r_{13} - r_{31}}{2 * \sin\theta} \qquad (5-4)$$

$$u_x = \frac{r_{21} - r_{12}}{2 * \sin\theta} \qquad (5-5)$$

$$\theta = \operatorname{acos}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \qquad (5-6)$$

$$r_{11} = \cos\alpha\cos\beta \qquad (5-7)$$

$$r_{12} = \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \qquad (5-8)$$

$$r_{13} = \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \qquad (5-9)$$

$$r_{21} = \sin\alpha\cos\beta \qquad (5-10)$$

$$r_{22} = \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma \qquad (5-11)$$

$$r_{23} = \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \qquad (5-12)$$

$$r_{31} = -\sin\beta \qquad (5-13)$$

$$r_{32} = \cos\beta\sin\gamma \qquad (5-14)$$

$$r_{33} = \cos\beta\cos\gamma \qquad (5-15)$$

5.1.3.2 Work object (build platform) configuration

Sometimes it is useful to configure a new work object (or feature) to create a new coordinates system on a plane. In this case, it is desired to have a new origin at the build platform (**Figure 5-10)** to locate the printing points. Universal Robots use the method of three points for the configuration of a new work-object: the new origin O_{new}, one point lying on the x-axis P_x, and one point lying on the y-axis P_y (**Figure 5-11**).

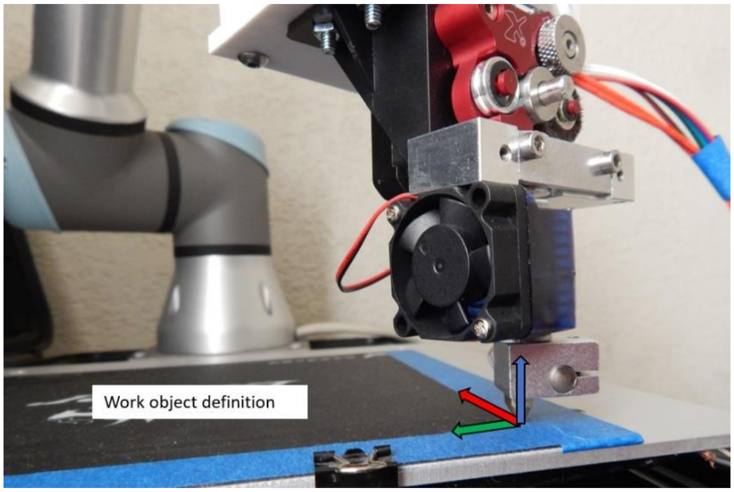


Fig. 5-10 Configuration of the build platform as a new work object to having a new origin with respect to the {Base} of the robot

The homogenous matrix for the new work object, in this case, the build platform, is needed for the transformation of each point of the printing trajectory (position of the extruder) with respect to the build platform and translated with respect to the base of the robot to move the TCP. So, the homogenous matrix is given by $(5-16)$ where the vectors $\vec{n}, \vec{s}, \vec{a}$ are defined in $(5-17)$ to $(5-19)$ (**Figure 5-9**).

$$^{0}A_{buildPlatform} = \begin{bmatrix} n_x & s_x & a_x & O_{new_x} \\ n_y & s_y & a_y & O_{new_y} \\ n_z & s_z & a_z & O_{new_z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (5-16)$$

$$\vec{n} = \frac{(P_x - O_{new})}{\|P_x - O_{new}\|} \qquad (5-17)$$

$$\vec{s} = \frac{(P_y - O_{new})}{\|P_y - O_{new}\|} \qquad (5-18)$$

$$\vec{a} = \vec{n} \times \vec{s} \qquad (5-19)$$

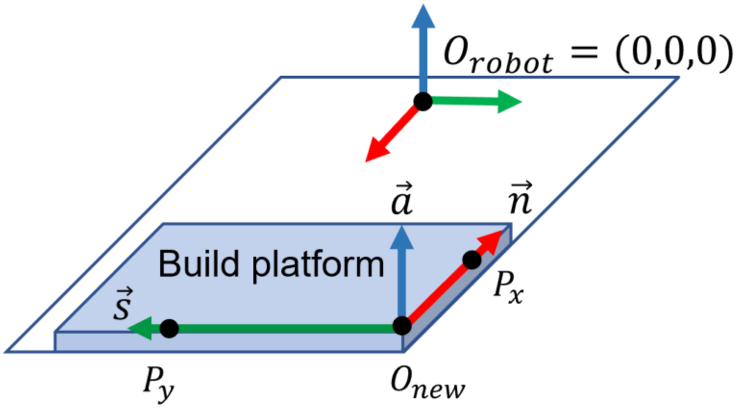


Fig. 5-11 Parameters acquired during the build platform configuration.

The resulting data representing the build platform cannot be visualized directly on the teach pendant, but it may be found inside the script file as the instruction $global\, mynewplane\ =\ pose$. As before, the *pose* is given as $p[x, y, z, \theta u_x, \theta u_y, \theta u_z]$, where (x, y, z) is the position of the new origin $[meters]$, and $(\theta u_x, \theta u_y, \theta u_z)$ is the orientation of the plane given in *axis-angle* notation $[radians]$. For instance, the following data in millimeters was shown on the teach pendant while the 3-point method was being followed to get the configuration of the build platform:

$$O_{new} = (448.4,121.91,100.71)$$
$$P_x = (313.68,122.1,91.58)$$
$$P_y = (447.04, -25.25,99.85)$$

The corresponding instruction found inside the script file was $global\, myPlane = p[0.44839, 0.12191, 0.10072, -0.10641, -0.00855, 3.13859]$, which means a position of the new origin located at $(x = 448.39mm, y = 121.91mm, z = 100.72mm)$ with an orientation given by the rotation vector $(\theta u_x, \theta u_y, \theta u_z) = (-0.10641, -0.00855, 3.13859)$ which corresponds to $roll = \gamma = -0.0055\, rad,\ pitch = \beta = 0.0678\, rad, yaw = \alpha = 3.1402\, rad)$

Analytically, the calculations to get the homogeneous matrix for the location and orientation of the build platform with respect to the $\{base\ =\ (0,0,0)\}$ of the robot can be obtained by doing the corresponding substitution using $(5-16)$ through $(5-19)$

$$\|P_x - O_{new}\| = \sqrt{(313.68 - 448.4)^2 + (122.1 - 121.91)^2 + (91.58 - 100.71)^2} = 135.029$$

$$\vec{n} = \frac{(-134.72, 0.19, -9.13)}{135.029} = (-0.9977, 0.0014, -0.0676)$$

$$\left\|P_y - O_{new}\right\| = \sqrt{(447.04 - 448.4)^2 + (-25.25 - 121.91)^2 + (99.85 - 100.71)^2} = 147.168$$

$$\vec{s} = \frac{(-1.36, -147.16, -0.0086)}{147.168} = (-0.0092, -0.9999, -0.0058)$$

$$\vec{a} = (-0.0092, -0.9999, -0.0058) \times (-0.0092, -0.9999, -0.0058)$$

$$= (-0.0676, -0.0052, 0.9977)$$

$$^{0}T_{buildPlatform} = \begin{bmatrix} -0.9977 & -0.0092 & -0.0676 & 448.4 \\ 0.0014 & -0.9999 & -0.0052 & 121.91 \\ -0.0676 & -0.0058 & 0.9977 & 100.71 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5.1.3.3 Mathematical model of the UR3/CB3

Using the Denavit-Hartenberg (DH) convention (Gries & Schneider, 2005; Murray et al., 2017), the parameters of the UR3 robot can be obtained in **Table 5-2** and shown in **Figures 5-12** and **5-13**. Using equations $(5-20)$ and $(5-21)$, the transformation matrix $^{0}T_6$ can be obtained.

Table 5-2 Denavit-Hartenberg parameters for UR3/CB3

Link	$\theta_i(rad)$	$a_i\ (m)$	$d_i\ (m)$	$\alpha_i(rad)$
1	θ_1	0	0.1519	$\pi/2$
2	θ_2	-0.24365	0	0
3	θ_3	-0.21325	0	0
4	θ_4	0	0.11235	$\pi/2$
5	θ_5	0	0.08535	$-\pi/2$
6	θ_6	0	0.08190	0

$$^{n-1}A_n = \begin{bmatrix} cos\ \theta_n & -sin\ \theta_n cos\ \alpha_n & sin\ \theta_n sin\ \alpha_n & r_n\ cos\theta_n \\ sin\ \theta_n & cos\ \theta_n cos\ \alpha_n & -cos\theta_n \sin\alpha_n & r_n\ sin\theta_n \\ 0 & sin\ \alpha_n & cos\ \alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & \boldsymbol{R} & & \boldsymbol{O} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5-20)$$

$$^{i}T_j = {}^{i}A_{i+1} \dots {}^{j-1}A_j = \begin{bmatrix} ^{i}\boldsymbol{R}_j & ^{i}\boldsymbol{O}_j \\ 0 & 1 \end{bmatrix} \quad (5-21)$$

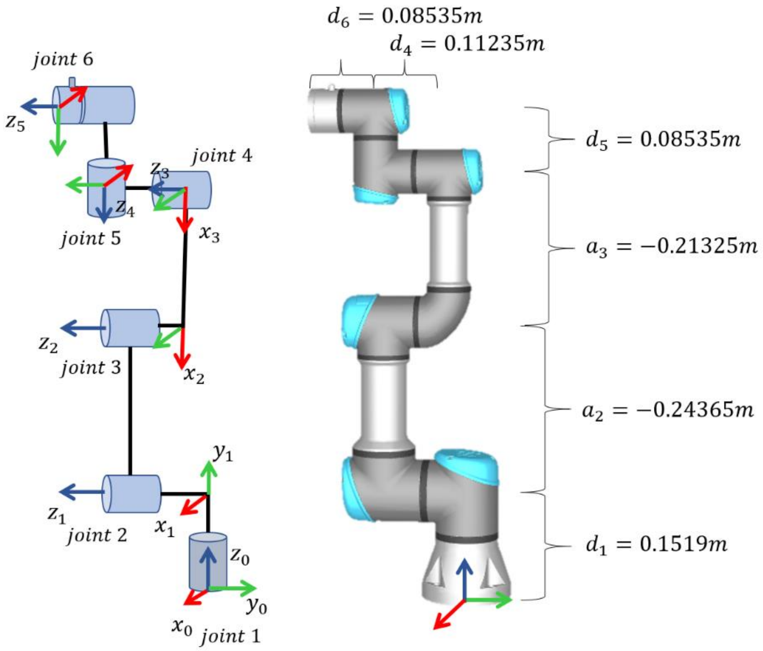


Fig. 5-12 Reference frames for DH analysis of robot UR3/CB3

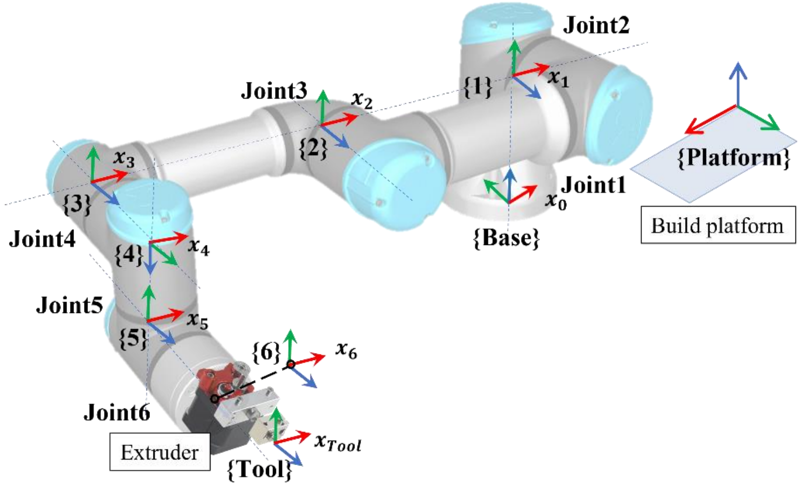


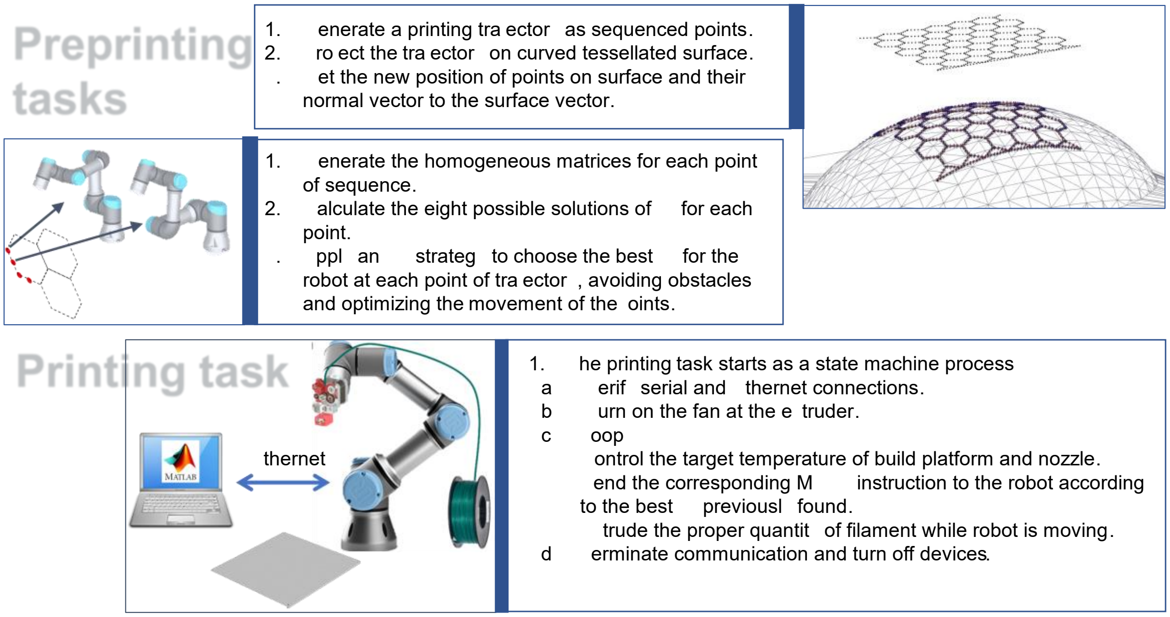
Fig. 5-13 The nomenclature is used according to the D-H convention to determine the joints' frames.

5.2 Complete software system for 3D printing

The block diagram of the software developed on the master Dell computer under Windows 10 and MATLAB 2021 for the development of this work is shown in **Figure 5-14** as it includes the following:

a) Software that is executed before the printing task: path and motion planning.

b) Software that is executed during the printing task: synchronization of tasks.



1. Generate a printing trajectory as sequenced points.
2. Project the trajectory on curved tessellated surface.
3. Get the new position of points on surface and their normal vector to the surface vector.

1. Generate the homogeneous matrices for each point of sequence.
2. Calculate the eight possible solutions of for each point.
3. Apply an strategy to choose the best for the robot at each point of trajectory, avoiding obstacles and optimizing the movement of the joints.

1. The printing task starts as a state machine process
   - a Verify serial and Ethernet connections.
   - b Turn on the fan at the extruder.
   - c Loop
     - Control the target temperature of build platform and nozzle.
     - Send the corresponding M instruction to the robot according to the best previously found.
     - Extrude the proper quantity of filament while robot is moving.
   - d Terminate communication and turn off devices.

Fig. 5-14 Flow diagram of the execution of the developed and implemented software.

5.2.1 Software that is executed before the printing task.

Before sending the MOVE instructions to the robot, the scripts in MATLAB 2021 for the following tasks need to be executed:

- Generation of the printing trajectories as sequenced points.’
- Projection of these points on a tessellated surface.
- Getting the new location of each point and its orientation normal to the surface where the point is lying.
- Calculation of the transformation matrices and the inverse kinematics (IK) of the UR3 for each point of the printing trajectory.
- Application of a Reinforcement Learning strategy for the optimization of the movements of the joints and obstacle avoidance.

5.2.1.1 Generation of the printing trajectories and their projection (path planning).

The first step is the generation of the continuous printing trajectory which may be generated on a plane using parametric equations, iterative/recursive functions, Lindenmayer systems, or any combination of them. The procedure is the same as explained in our work (Rodriguez-Padilla et al., 2021), where printing trajectories such as re-entrant, hexagonal, and Hilbert have been generated using Lindenmayer systems and parametric equations.

Once the sequenced points (x_i, y_i, z_i) of the printing trajectory are generated, and the algorithm of vectorial projection is used to relocate each point to the tessellated curved surface to achieve the deposition of material conformally. The algorithm gives as output the new location of each point on the curved surface and its normal vector $(x_i', y_i', z_i', n_x, n_y, n_z)$ as indicated in **Figure 5-15**.

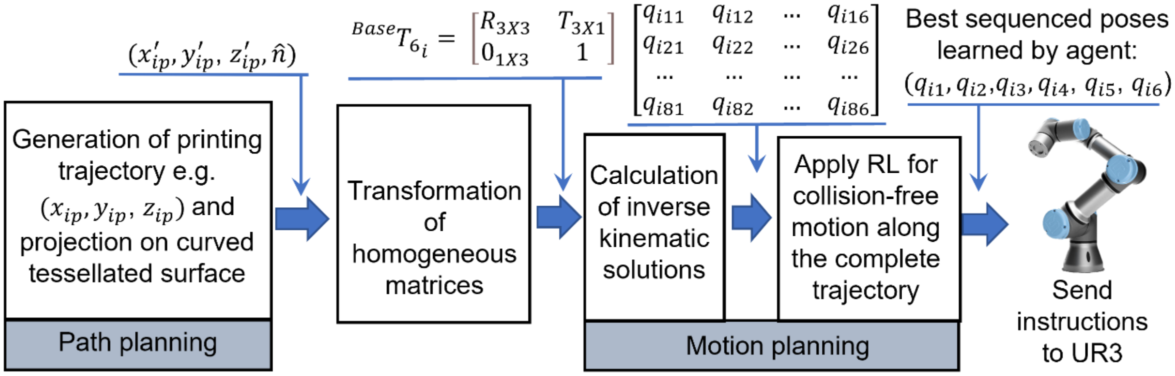

$(x_{ip}', y_{ip}', z_{ip}', \hat{n})$
${}^{Base}T_{6_i} = \begin{bmatrix} R_{3X3} & T_{3X1} \\ 0_{1X3} & 1 \end{bmatrix}$
$\begin{bmatrix} q_{i11} & q_{i12} & \cdots & q_{i16} \\ q_{i21} & q_{i22} & \cdots & q_{i26} \\ \cdots & \cdots & \cdots & \cdots \\ q_{i81} & q_{i82} & \cdots & q_{i86} \end{bmatrix}$
Best sequenced poses learned by agent:
$(q_{i1}, q_{i2}, q_{i3}, q_{i4}, q_{i5}, q_{i6})$
Generation of printing trajectory e.g. (x_{ip}, y_{ip}, z_{ip}) and projection on curved tessellated surface
Path planning
Transformation of homogeneous matrices
Calculation of inverse kinematic solutions
Apply RL for collision-free motion along the complete trajectory
Motion planning
Send instructions to UR3

Fig. 5-15 Flow diagram of path and motion planning for robot UR3

5.2.1.2 Calculation of the transformation matrices.

It is important to mention that the obtained data points of the printing trajectory (x_i', y_i', z_i') are located with respect to the build platform which was previously configurated with respect to the {Base} of the robot. Here, the matrices of transformation are used to get the location and pose of each printing point with respect to the {Base} of the robot taking into consideration the extruder as a tool. The reason for this is due to the instruction MoveJ (Move Junction) will be used, rather than MoveL (Move Linear), to move the robot at each position. The following steps are executed:

1) Using the manipulator's arm, configure the build platform as a local coordinate frame to form ${}^{Base}T_{Platform}$
2) Using the manipulator's arm, configure the TCP to form ${}^{6}T_{Tool}$
3) For each point of the printing trajectory,
    - Get ${}^{Platform}T_{Tool}$ from $(x_i', y_i', z_i', n_x, n_y, n_z)$

      For the calculation of the extruder's orientation, given that the robot is redundant since only five degrees of freedom are needed for printing (the extruder is rotationally symmetric), its $\overrightarrow{x_{Tool}}$ axis coincides with the x-axis of the build platform while its $\overrightarrow{y_{Tool}}$ and $\overrightarrow{z_{Tool}}$ vectors are dynamically calculated according to the normal data vector of each trajectory point.

- Compute ${}^{Base}T_{Tool} = {}^{Base}T_{Platform}\,{}^{Platform}T_{Tool}$
- Compute ${}^{Base}T_{6} = {}^{Base}T_{Tool}\left({}^{6}T_{Tool}\right)^{-1} = \begin{bmatrix} R_{3X3} & T_{3X1} \\ 0_{1X3} & 1 \end{bmatrix}$

5.2.1.3 Inverse kinematics for the manipulator's arm UR3

To avoid obstacle collisions and propose a strategy for optimization, the inverse kinematics of the UR3 needs to be obtained.

The inverse kinematics (IK) problem for the manipulator's arm can be solved using approximated numerical methods or exact methods. In this work, the exact method was preferred, and it is based on the work presented by (Abdelaziz et al., 2019) for a robot UR3. The analysis of the paper was verified theoretically and physically, and several adjustments were implemented since some mismatching was detected. Having ${}^{Base}T_{6}$, up to eight different possible solutions for the IK for the robot UR3 may be available for each point of the printing trajectory as represented in **Figure 5-16**.

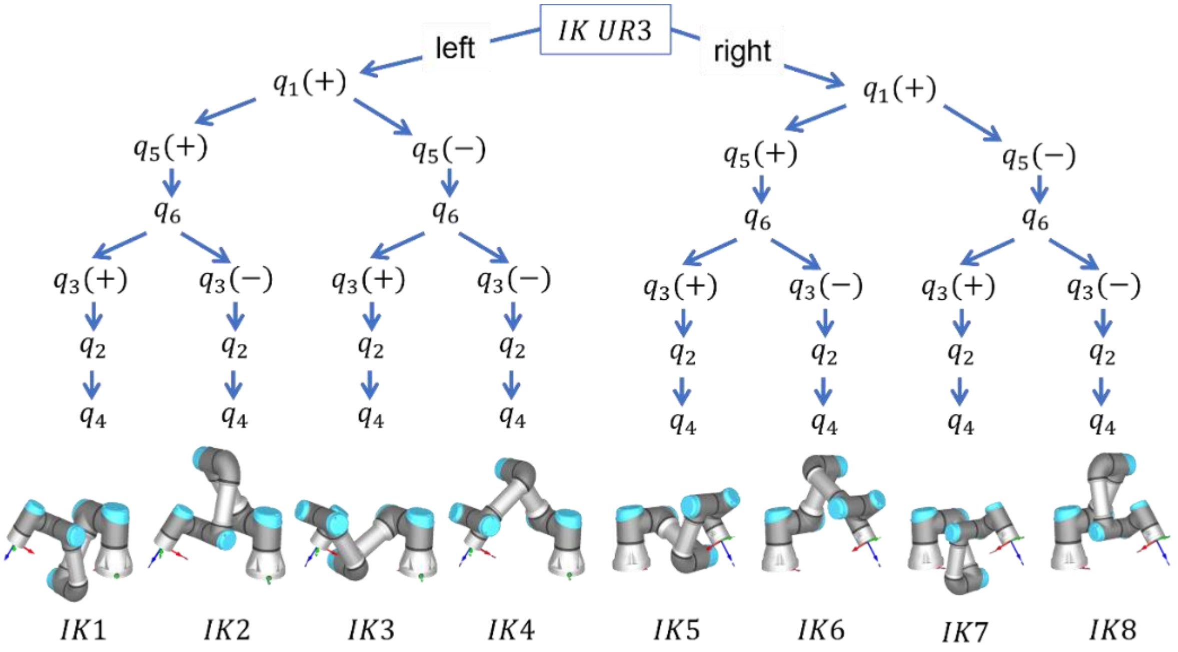


Fig. 5-16 Up to eight solutions for the inverse kinematics of UR3

5.2.1.4 Application of a Reinforcement Learning (RL) strategy for optimization.

According to (Sutton et al., 1998; Sutton & Barto, 2018), the idea behind RL was initially simple and obvious: follow the process of learning things in the same way that humans do by trial and error. However, translating the initial idea into a mathematically based theory was a difficult process, which was successfully formulated using the Markov

Decision Process (MDP) to describe mathematically the interaction between an agent and its environment (through a tuple of five elements $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$), and using Dynamic Programming (DP) and Monte Carlo methods to solve MDP. In this manner, RL considers an intelligent agent learning how to make good (optimal) sequences of decisions under uncertainty. Sequential decision-making consists of an iterative closed-loop process where at each step *t*, an intelligent *agent* gets a representation of the environment *state* $s_t \in \mathcal{S}$, takes an *action* $a_t \in \mathcal{A}$ that modifies the *environment* (initially unknown), moves to the next state $s_{t+1} \sim \mathcal{P}(s_t, a_t)$ and obtains a reward $\mathcal{R}_t$ (scalar), as represented in **Figure 5-17**. The goal is the selection of the actions that maximize the cumulative expected future reward $G_t = \sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{t+k+1}$, where γ represents the discount factor weights for future rewards $(0 \leq \gamma \leq 1)$. The way the agent chooses an action is determined by the policy $\pi(a|s)$ which maximizes the expectation of the sum of discounted rewards and maps from states to actions.

Here, the implementation of standard RL by policy improvement is presented, to find an adequate collision-free pose configuration for the robot following the planned printing trajectory. It is model-free based, which means it learns by trial-and-error and it does not use a model of the environment dynamics, on the deterministic policy gradient. The actions are turning the motors for each of the joints clockwise or counterclockwise to achieve one of up to eight IK solutions. The states consist of up to eight different possible configurations of the inverse kinematic of the robot $\mathcal{S} = \{s_{1,} s_2, \ldots s_8\}$ where $s_i = \{q_1, q_2, \ldots q_6\}$. The reward function is related to the two important aspects of optimality defined by $(5-22)$ and $(5-23)$: a) the sum of the difference in rotation angles between pose n and pose $(n+1)$ since a greater difference involves a longer time for the robot to achieve the next pose impairing the quality of the printing; b) the distance between the main joints of robot and the obstacles such as the build platform. Obstacles are analytical-defined functions (e.g., a plane or sphere) considering the worst case when d_{obs}, the distance between the main joints of the robot and each obstacle, is the shortest.

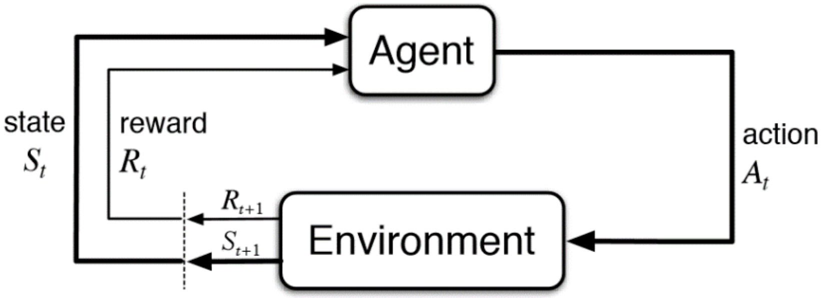


Fig. 5-17 General scheme for a standard RL strategy.

The RL algorithm by policy improvement (Sutton & Barto, 2018) involves first the iterative loop for the policy evaluation as the calculation of the state-value function v_π, for an arbitrary policy π for all $s \in \mathcal{S}$, where $\pi(a|s)$ is the probability of taking action a in state s under policy π. Then, the iterative policy improvement by finding a better (greedy) policy given by equation (2)

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s',r|s,a)[r + \gamma v_\pi(s')] \quad (5-22)$$

$$\pi'(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} \sum_r p(s',r|s,a)[r + \gamma v_\pi(s')] \quad (5-23)$$

Here, the agent initially follows the random policy according to the IK solutions (up to eight) considering equal probabilities (e.g., eight IK possible solutions correspond to $\pi(a|s) = 1/8$)

5.2.1.5 Defining the reward functions

Given N points of a trajectory, the manipulator arm may have up to eight inverse kinematics different solutions $IK_{1\le i\le 8}$ at each point n, and this pose $IK_{(i,n)}$ involves an angular value $q_{(j,i,n)}$ for each joint j, where $1 \le j \le 6$ for 6 DoF. Since the robot takes time to reach the next pose $IK_{(i,n+1)}$ going from point n to point $(n+1)$, which may impact the quality of the printed piece, the first reward $R_{joints(i,n)}$ is defined by the difference $(IK_{(i,n)} - IK_{(i,n+1)})$ given by $(5-24)$ and $(5-25)$. These differences are sorted to assign the rewards going from -1 (best) up to -8 (worst): a small difference of movement in joints is a good reward (tending to zero) meaning better maintenance of the printing speed. On the other hand, a large difference means up to a minimum reward of -8. The rewards are accumulated at each step n to achieve an optimization along the whole trajectory in addition to the optimal IK solution at each step n. A non-valid IK solution gets -1000 of reward.

$$\Delta q_{(j,n)} = \min\left(2\pi - abs\left(q_{(j,n)} - q_{(j,n+1)}\right), abs\left(q_{(j,n)} - q_{(j,n+1)}\right)\right) \qquad (5-24)$$

$$R_{joints(i,n)} = \sqrt{\sum_{j=1}^{6}(\Delta q_{(j,n)})^2} \qquad (5-25)$$

The second reward concerns obstacle avoidance, and it is defined by $(5-26)$ and $(5-27)$. The main aim is to avoid any collision with the build platform when printing is in progress considering the position of Joint3 and Joint4 at each $IK_{(i,n)}$ as the most crucial for this purpose. The reward is defined in general for any obstacle, as the shortest Euclidean distance between the position $(x_{joint3}, y_{joint3}, z_{joint3})$ and all M defined obstacles, in addition to the shortest Euclidean distance between $(x_{joint4}, y_{joint4}, z_{joint4})$ and all M defined obstacles. A large distance between joints and obstacles means a good reward (-1). On the other hand, a small distance means almost a collision gets a maximum penalty (up to -8). Any collision gets -1000 of reward. Obstacles may be represented by a plane and/or a geometric body defined by an equation (e.g., sphere). The location in space of Joint3 and Joint4 $(x_{jointK}, y_{jointK}, z_{jointK})$ are calculated using the Denavit-Hartenberg (D-H) transformation matrices. The second reward is obtained after sorting the distances, while the total reward is finally defined by the sum of the first and second rewards $\mathcal{R}_{total} = R_{joints} + R_{obs}$. The last point of the printing trajectory works as a target point having a reward equal to 0 (the best reward) for all the possible IK solutions. The global optimization objective consists of the $\min \Delta q_{(j,n)}$ while $\max|JointK - obstacle_m|$

$$|JointK - obstacle_m| = \sqrt{\left(x_{jointK} - x_{obs_m}\right)^2 + \left(y_{jointK} - y_{obs_m}\right)^2 + \left(z_{jointK} - z_{obs_m}\right)^2} \quad (5-26)$$

$$R_{obs(i,n)} = \min(\min\,(\|(Joint3 - obstacles\|), \min\,(\|(joint4 - obstacles\|)) \qquad (5-27)$$

5.2.2 Software during printing task

Once the data containing the sequence of the inverse kinematics best solution of the UR3 for each point through the complete printing trajectory, the next step is to run the state-machine code implemented in a Dell Inspiron 15 using Windows 10 and MATLAB 2021. This state-machine code has the following main tasks:

- Initialization of serial and Ethernet communication.
- Following a state machine for the control of the temperature of the nozzle and build platform, synchronization of the movements of the joint of the robot with the precise rotation of the stepper motor for extrusion of the filament.
- Finishing the printing process by turning off all the devices and terminating communications.

5.2.2.1 State machine to control the system

The script running in MATLAB 2021 has the main function of controlling the Arduino-based board as a slave through the handshaking of commands sent by the serial port. The main tasks of the master system are:

- Generate the printing trajectories.
- Calculate the position of the extruder with respect to the base of the robot for each point of the trajectory.
- Calculate the inverse kinematics (IK) for each point of the trajectory.
- Uses a Reinforcement Learning strategy to find the optimal IK solution.
- Send instructions to move the robot.
- Get information from the robot about its current status.
- Send the needed commands to the Arduino-board for
    - Set the desired temperature for the extruder and build the platform.
    - Get measurements from temperature sensors for display.
    - Start/stop the process of temperature control of the extruder and the build platform.
    - Turn on/off the fan
    - Calculate the number of steps for the motor equivalent to the quantity of filament to be extruded.

Figure 5-18 shows the state machine implemented to control the functionality of the whole master system, where the main states have been indicated.

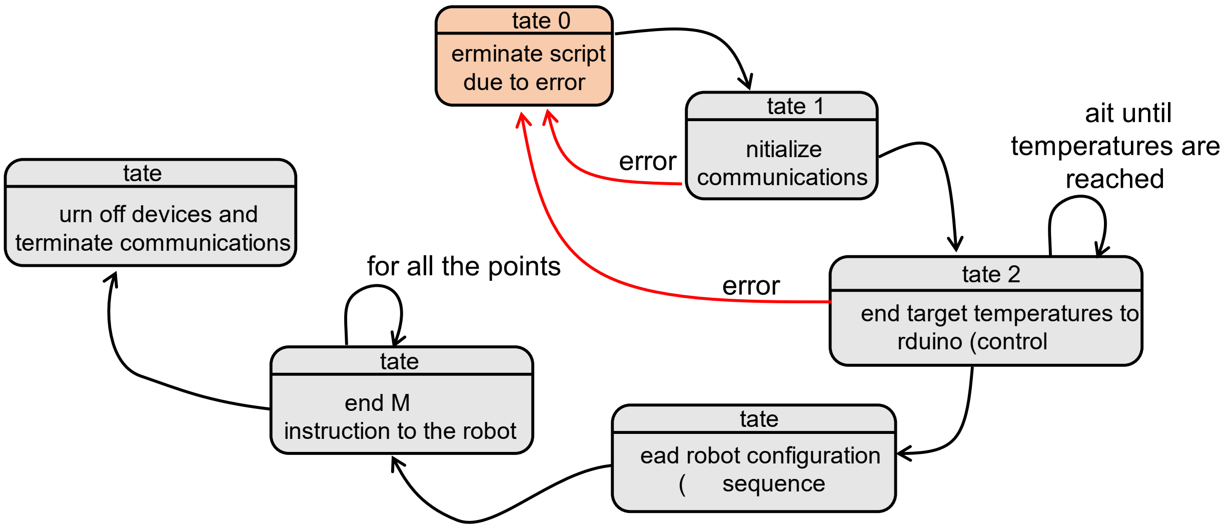

tate 0
erminate script due to error
tate 1
nitialize communications
error
ait until temperatures are reached
tate
urn off devices and terminate communications
for all the points
error
tate 2
end target temperatures to rduino (control
tate
end M instruction to the robot
tate
ead robot configuration (sequence

Fig. 5-18 State machine implemented in the master system.

First, the master initializes the serial and Ethernet communications. The target temperatures of the nozzle and build platform are defined by the user and sent to the Arduino which is responsible for maintaining control of both temperatures. The master read the dataset that contains the optimized inverse kinematics sequenced for each of the printing points. The master prepares the instruction MOVEJ for the robot UR3 and it is sent through Ethernet one by one assuring the robot is reaching the pose at each step. Finally, the communications are terminated, and all the devices are turned off. In case of error, everything goes to State 0.

Chapter 6 Results obtained

In this chapter, the results obtained after the implementation of a complete exercise for the validation of the proposed framework are presented in detail.

6.1 Validation of the strategy for the conformal deposition of fused filament

To validate the effectiveness of RL for obstacle avoidance of the robot (Joint3 and Joint4), and the optimization of the total angular displacement of the joints, a printing trajectory was projected on a non-planar tessellated surface having a total of 291 points as shown in **Figure 6-1**. Three different tests were executed: first, the simulated performance of angular displacement of each joint for the eight different IK solutions without obstacles, and the comparison with the optimal motion planning learned by the RL agent; second, the setting of different obstacles and the observation of the collision-free motion planning learned by the RL agent; third, the actual implementation of the complete strategy using the robot UR3 to print a lattice sample.

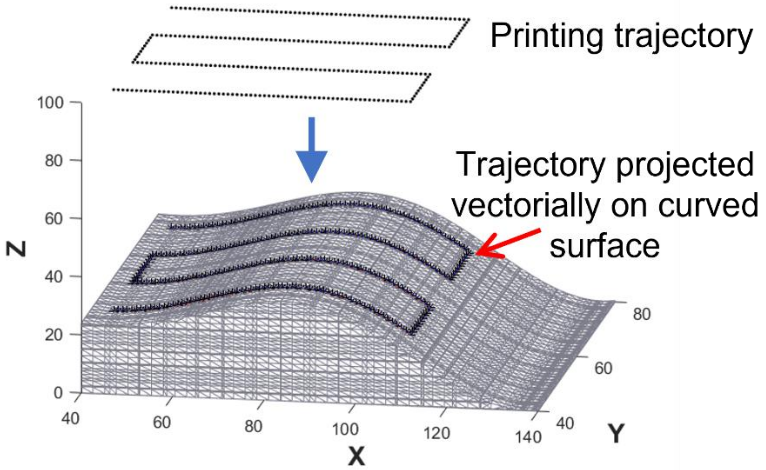


Fig. 6-1 Printing trajectory projected on a curved surface.

6.2 RL agent solution with no obstacles

The strategy used for Universal Robots® to decide which IK solution to choose for the next pose of the end effector is based on the current or last pose. In this context, to validate the set of poses chosen by the RL agent, the accumulated angular displacement of all joints ($\sum_{n=1}^{291}\sum_{i=1}^{6}\Delta q_i^n$) from pose to pose along the complete trajectory selecting the IK1 solution was calculated ($IK1 = 2137.98\ radians$). The same was done by selecting

IK2 $(IK2 = 2392.86\, radians)$, IK7 $(IK7 = 1992.25\, radians)$, and IK8 $(IK8 = 2258.67\, radians)$ solutions since IK3, IK4, IK5, and IK6 were not always an available solution for the curved printing trajectory of this example, as shown in **Figure 6-2**. So, the order of optimality would be IK7, IK1, IK8, and IK2; and after running the RL algorithm, the optimal solution for the complete trajectory without obstacles was given as the IK7 for a prediction factor $\gamma = 0.95$, just as expected.

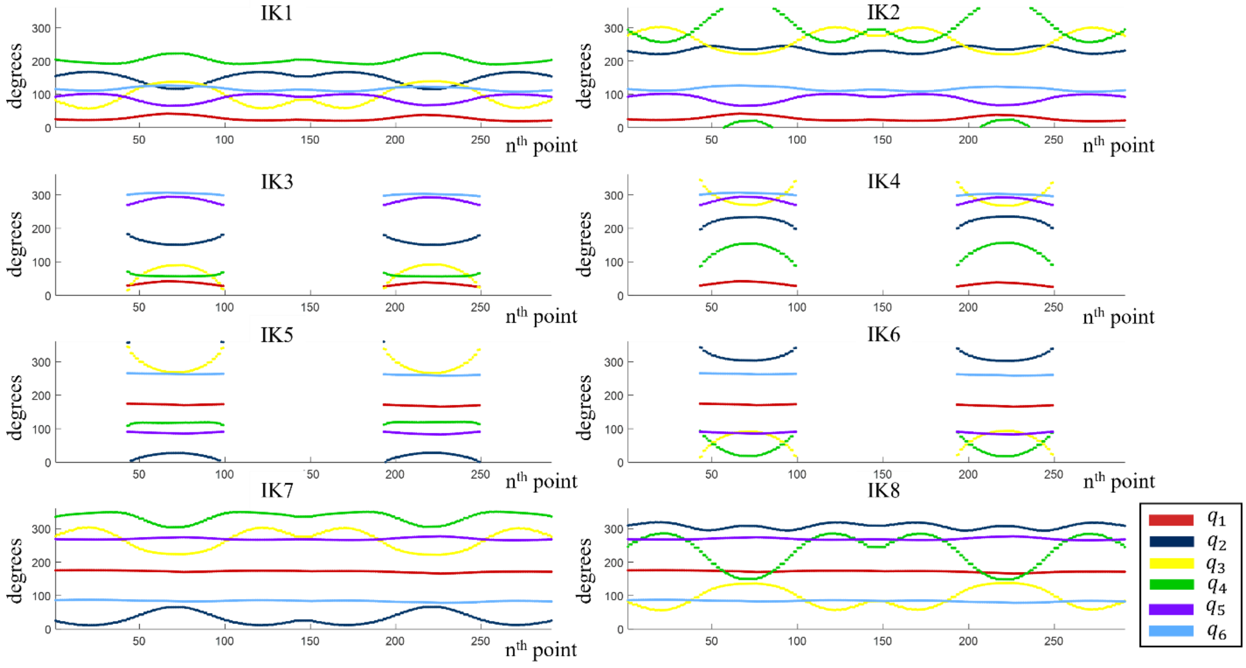


Fig. 6-2 Behavior of each joint q_i for the different inverse kinematic solutions.

6.3 Obstacles avoidance

As explained in the previous section, the optimal IK solution without obstacles is IK7. For the validation of obstacles avoidance, three different tests were executed: 1) inserting a plane as an obstacle (similar to having a build plane), hence IK7 should stop being optimal; 2) inserting also a sphere as a second obstacle, and 3) inserting several spheres distributed into the working space of the robot. First, a plane at $z = 30mm$ was inserted as shown in **Figure 6-3**, where the agent found that following the IK8 is the optimal solution to achieve minimal joint displacement and avoid the collision of Joint3 which does occur with IK1 and IK7, while IK2 has the worst behavior $(IK2 = 2392.86\, radians)$ as already shown in **Figure 6-2**. Then, keeping the plane as an obstacle, a sphere of radius $20\, mm$ was inserted as a second obstacle at $(X =$

$150mm, Y = -50mm, Z = 350mm)$ trying to invade the track followed by joint 3 when IK8. **Figure 6-4** shows the chosen trajectory after running the RL strategy, from point 1 to point 42, the robot follows IK7; then from point 43 to point 291 follows IK2.

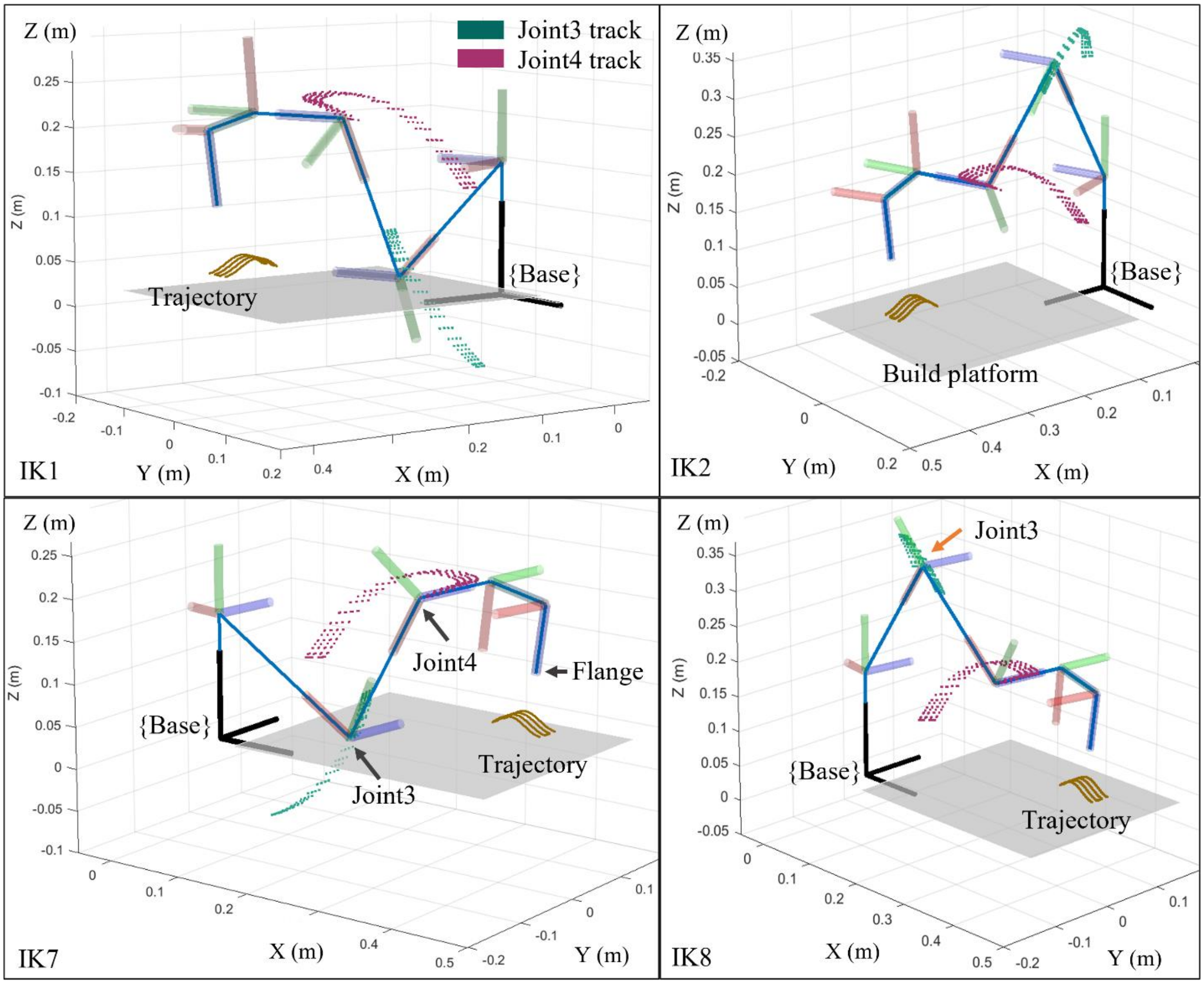


Fig. 6-3 The IK solutions available for a printing trajectory where the tracks for Joint3 and Joint4 are shown.

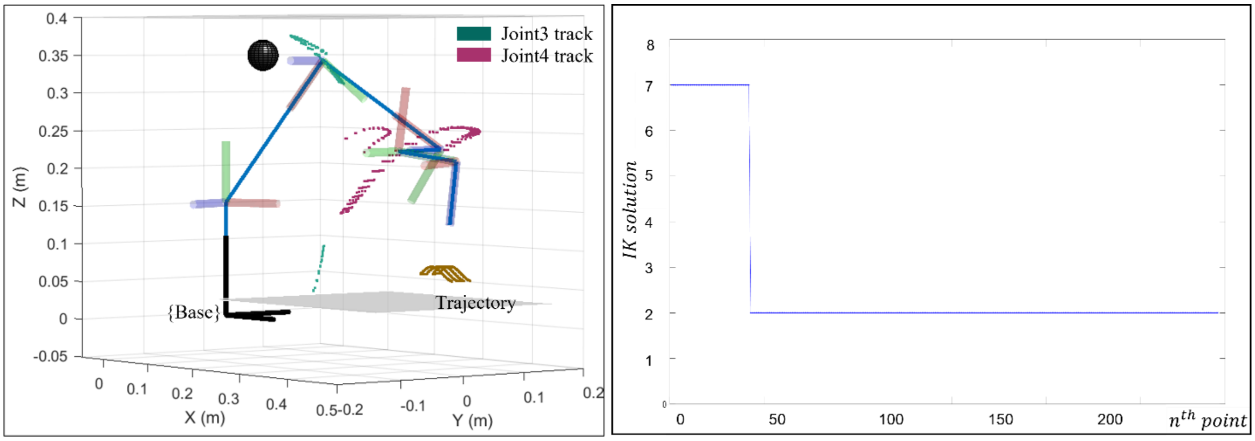


Fig. 6-4 RL strategy learned by the agent to evade the obstacles: a plane at $z = 30mm$, and a sphere inserted as an obstacle.

A third test is carried out where the IK solutions for the printing trajectory can be observed in detail: the behavior of the track of Joint3 for IK1 is opposite to that for IK2 while the track for Joint4 is the same (**Figure 6-5**). Similarly, the track of Joint 4 is the same for IK7 and IK8 while the tracks of joint3 are opposites. In this test, an obstacle, a sphere of radius $70mm$ located at $(200mm, -25mm, 0)$, is introduced trying to block the track of joint 3 of IK7, and after running the RL algorithm, the agent learns that following IK2 is the optimal solution since the positions of Joint3 and Joint4 for IK2 are the furthest from the obstacle, meaning a total better reward, even though IK2 does not get the best reward due to the joints displacement. Under this previous environment, the second sphere of radius $70mm$ and located at $(140,80,350)$ was inserted trying to block Joint3 of IK2 and the optimal solution learned by the RL agent was IK1 as shown in **Figure 6-6**.

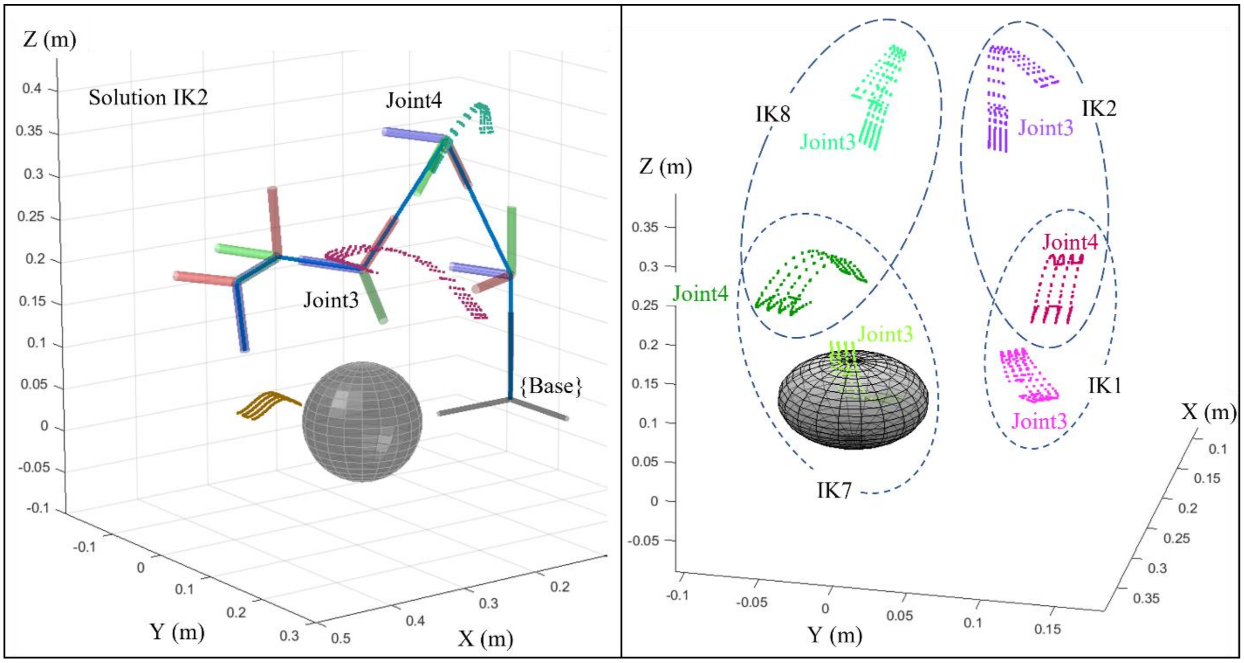


Fig. 6-5 Behavior of the tracks for Joint3 and Joint4 for the available IK solutions and their relationship.

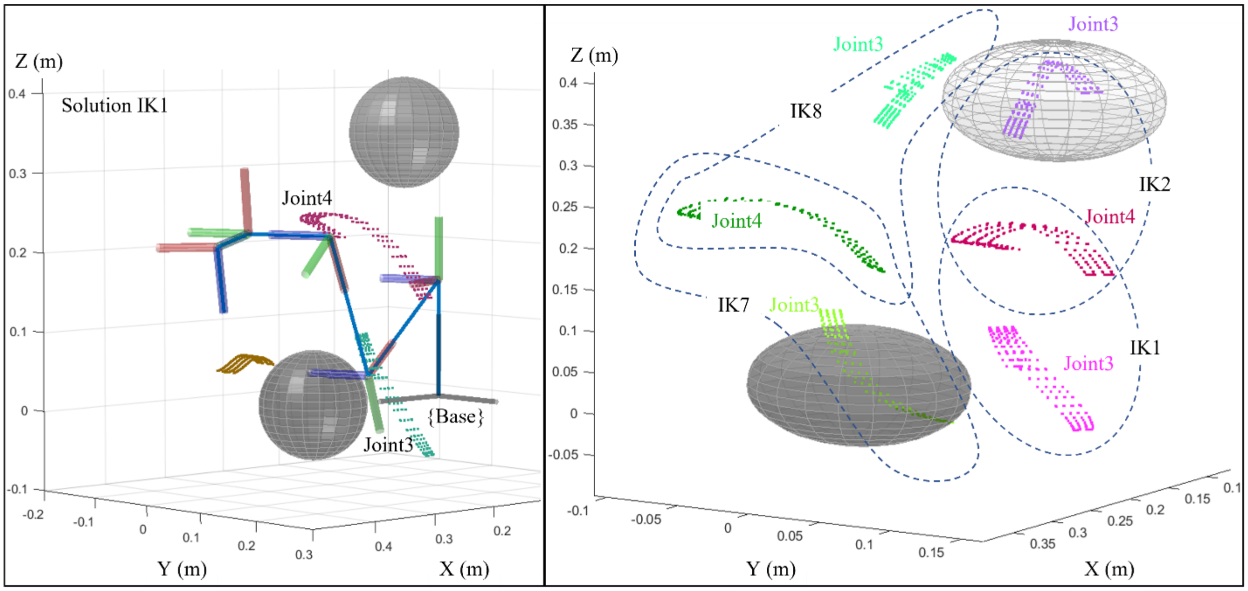


Fig. 6-6 IK1 solution chosen by the RL agent where an obstacle is blocking IK2, and a second obstacle is blocking IK7.

6.4 Using a manipulator arm for printing a lattice sample

The third validation was the implementation of the complete system including hardware and software for the fabrication of a lattice sample. The RL strategy considers the build platform as an obstacle.

6.4.1 Experimental setup

The complete system is shown in **Figure 6-7** and **Figure 6-8** as explained in detail in chapter 5, which consists of a robot UR3, a fixed build-platform with heater element and temperature sensor, Arduino board (Mega and RAMPS board, and 12V power supply), the Flexion extruder mechanism (heater, temperature sensor, fan, stepper motor, and nozzle of $0.4mm$) attached to the robot, and a network interface device. MATLAB 2020 is used as the programming platform and acts as the master controller. The Arduino board acts as a slave under the main control of the master, which uses a customized communication protocol to send the commands to be executed. These commands send the desired temperature for the platform and nozzle, starting/stopping the control process of temperature, extruding X quantity of filament by rotating the stepper motor N steps, getting the current temperature of nozzle and platform for display, starting/stopping the printing process and so on. The printing process starts through a state machine when the master sends the initial settings (desired temperature of nozzle and build platform) to the slave, and it starts controlling the temperature of the nozzle and the build platform. After reaching both target temperatures, the printing process begins by sending instruction by instruction to the robot (via 125Hz high-speed Ethernet) corresponding to each projected point of the printing trajectory.

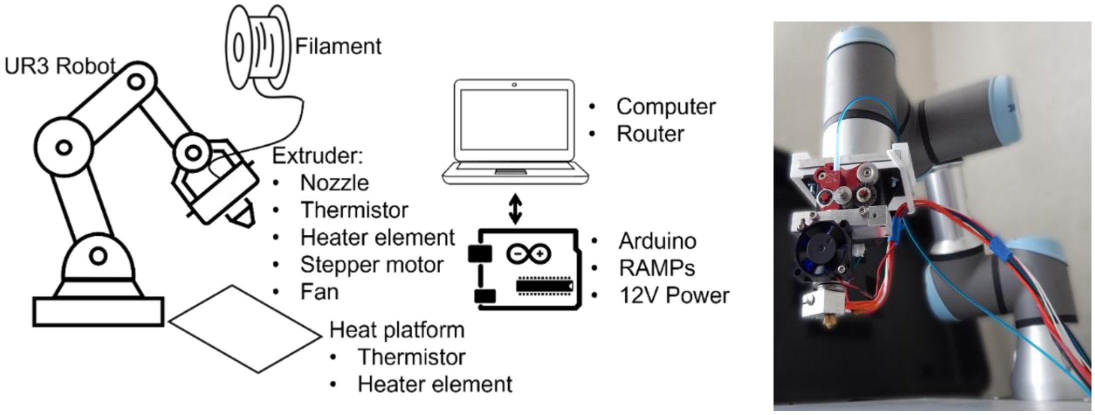


Fig. 6-7 a) Block diagram of the complete system and b) the actual UR3 with Flexion extruder attached.

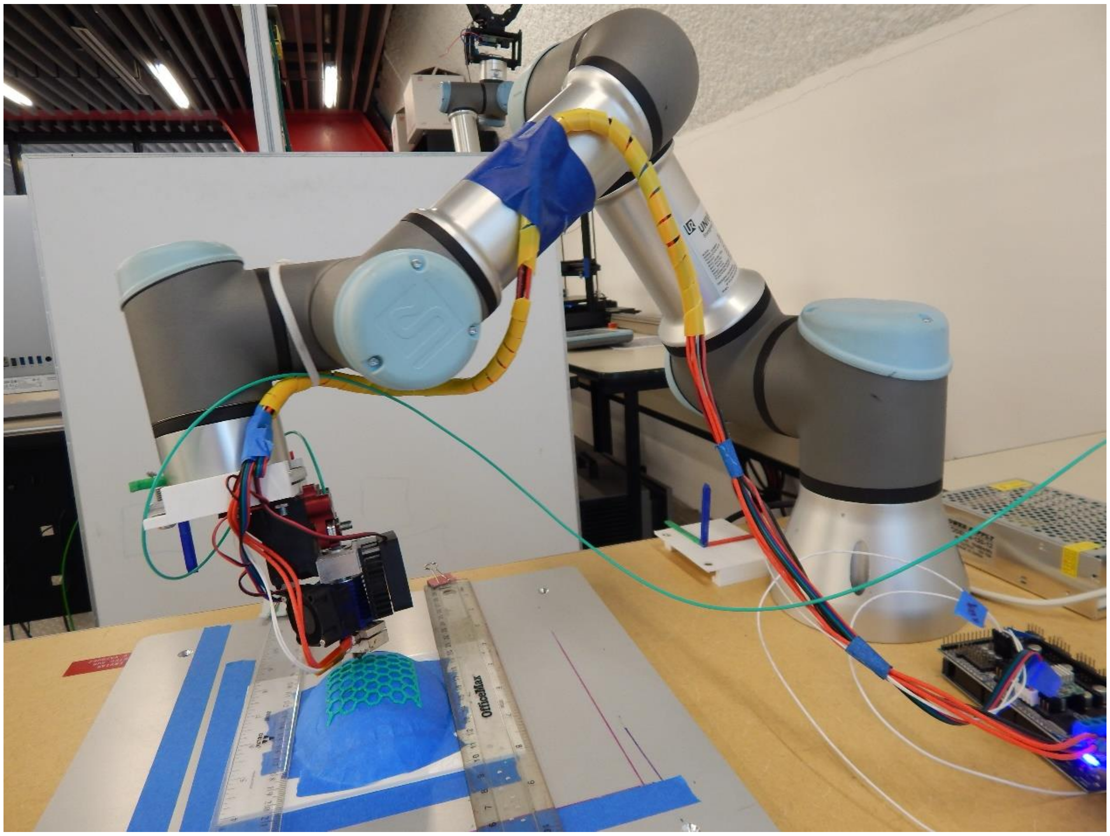


Fig. 6-8 Actual system implemented in the laboratory.

This actual printing test consisted of the projection of a re-entrant trajectory on a non-planar surface having a total of 1569 points as the printing trajectory. After running the RL algorithm having the build platform defined as an obstacle, the RL strategy successfully works to avoid the collision of the Joint3 and Joint4 of the robot and the platform. Even though the RL strategy is to find optimal IK, every instruction sent to the robot introduces a transmission delay in addition to the already known delay while reaching the next pose, producing an accumulation of material. To improve the quality of the sample, a strategy was implemented while the printing process is running that consisted in pausing the extrusion of material, sending only data concerning the pose of the robot, sending data concerning the translation, and resuming the extrusion of material. In this manner, the accumulation of material while the robot is reaching its next pose can be reduced and printing quality improved. **Figure 6-9** shows the re-entrant lattice printed on a curved tessellated surface using the proposal system. The complete lattice was made of three layers at steps of $0.2mm$ in the z-axis, using PLA as material and a nozzle of $0.4mm$.

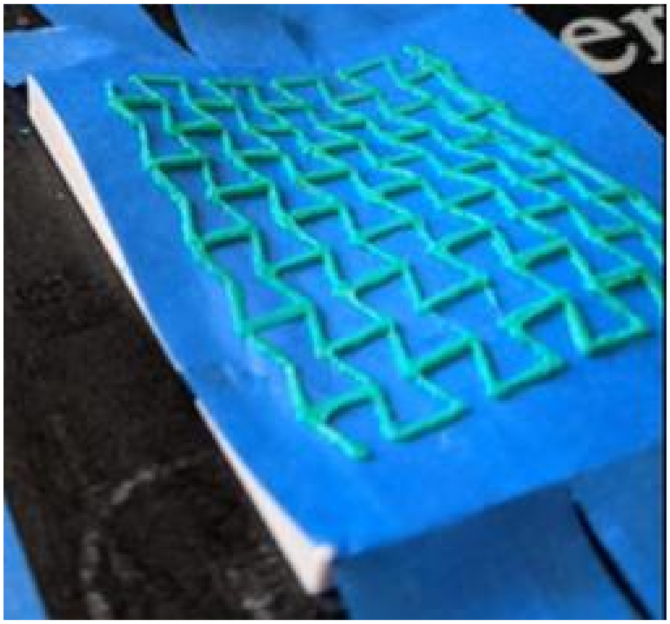


Fig. 6-9 Re-entrant pattern printed on a curved surface using UR3.

Figure 6-10 shows another example of the lattice printed on another curved tessellated surface, while the corresponding video of the actual test can be reviewed at https://youtu.be/njW20uMbndk

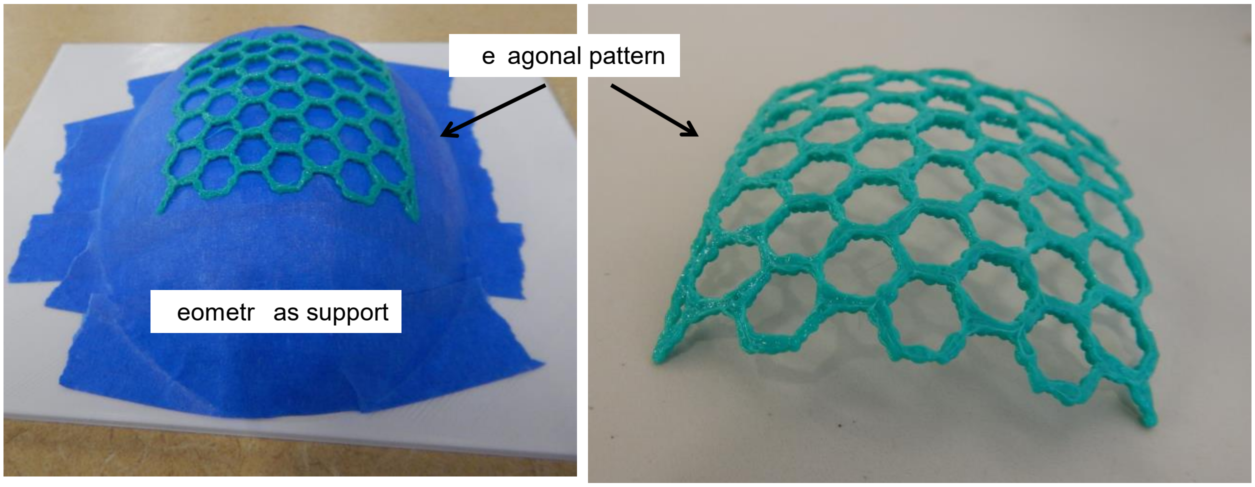


Fig. 6-10 Another example of conformal printing as a hexagonal lattice.

6.5 Discussion

According to the first results obtained **(Figure 6-3)**, it can be observed how the RL strategy effectively finds IK7 as the optimal solution for the whole printing trajectory without any obstacle, which corresponds to the minimum accumulated displacement of joints according to the available solutions $\{IK1 = 2137.98\, rad, IK2 = 2392.86\, rad, IK7 =$

$1992.25\,rad, IK8 = 2258.67\,rad\}$. The insertion of a plane (platform) as an obstacle produces the IK7 stop being the best option due to the collision of Joint 3. Here, the RL agent chooses IK8 as the optimal solution, since IK1 also produces collision and IK2 represents the worst choice of all. When a sphere is also added in this setting as a second obstacle trying to block the track followed by IK8 (**Figure 6-4**), it can be observed that IK7 start being optimal, but at some point, it receives penalties due to the collision giving as a result that the RL agent first chooses IK7 and then properly changes to IK2. The third simulated test allows observing in detail how the IK solutions relate to each other and how the RL agent chooses properly the optimal solution when certain IK receives penalties due to the collisions at some points. After testing the RL strategy for obstacle avoidance, the intelligent agent effectively learns the best motion sequence of poses when inserting different obstacles. Although the applicability for FDM here was to avoid collision only with the build platform, the RL strategy may be used even in real-time to avoid collisions while the object being printed grows over time or is surrounded by obstacles.

The final validation of the complete system was after the successful projection of a complex printing trajectory on a tessellated curved surface, then matrix-transformed to apply the inverse kinematic solutions and an RL strategy to maintain speed printing and avoid collision with the build platform. It is important to mention how the success of any FDM process depends on some technical factors such as the size and quality of the extruder. On the other hand, the literature to the best of our knowledge does not mention how to achieve the synchronization between the robot and the printing mechanism. Here, a state machine was implemented having full control of motors (extruder and fan), and the implementation of a PID temperature control (nozzle and build platform). The commands for the robot were generated one by one, to be sent through a client-server framework in a closed loop waiting for the robot to reach the target by reading its current position. This causes some unavoidable delays especially when a significant rotation is involved, and this alters the desired printing speed causing low-quality printing results. However, here a strategy to counteract these problems was proposed to reach first the next orientation of the robot while no filament is extruded, and then the robot is translated.

Chapter 7 Conclusions

This work presents the problem statement found after the review of the literature, finding a gap in work for the achievement of conformal curved printing on both analytical and tessellated surfaces for complex printing trajectories. Three different alternatives for the generation of printing trajectories were suggested: 1) using parametric equations, 2) using L-systems, and 3) from the data contained in a readable text file.

As a first significant contribution, it can be said that the algorithm developed to project (using the direction of a user-defined vector) any sequenced and ordered points to any tessellated surface, obtaining the new position of each point on the surface and the surface-normal vector used to orient any tool in a multi-axis machine.

This work also presents a complete system, procedure, and strategy for the achievement of the deposition of material on curved surfaces using a manipulator arm UR3 and an RL strategy for the optimization of angular movements of the robot and obstacle avoidance. Some simulations were executed to verify the effectiveness of the RL strategy using a simple zig-zag printing trajectory projected on a curved surface where a collision occurs for some IK solutions, and the incorporation of obstacles surrounding the robot space. Similarly, the actual printing of a re-entrant pattern projected on a curved surface was successfully executed using a UR3 robot for the achievement of higher degrees of freedom. The RL strategy applied in robotics for FDM shows promising results since it allows optimization during the execution of the printing task. One of the difficulties, when RL is applied, is the definition of the rewards functions since the convergence of the algorithm depends on that. Here, the proposal of the rewards functions for the optimization of the motion planning of the robot while obstacles are avoided has been successfully implemented.

In this context, the main contributions of this work may be established as

- Following our first work (Rodriguez-Padilla et al., 2021), where was presented an algorithm to map complex continuous trajectories, such as those patterns found in lattices, to any tessellated curved surface, where the work is translated to be implemented using a manipulator arm UR3 for the achievement of actual printing.

- Use of a Reinforcement Learning strategy for improving the printing quality by the optimization of the movement of the joints of the manipulator's arm, and obstacle avoidance.
- Proposal of reward functions for the optimization of the movement of joints and obstacle avoidance.
- Proposal of a strategy that helps to maintain the printing speed to counteract the delays due to the transmission/reception of data when the robot is controlled wireless.
- Implementation of a complete system using a manipulator arm (UR3) for the achievement of actual curved printing.

As a future work, it was detected several opportunities such as

1) The improvement of the quality of the printing task by implementing a better strategy way to reduce the time elapsed between sending the Move instruction to the robot and the reception of the next Move instruction. The problem here was that after sending the Move instruction, the state machine must wait until the robot reaches the target position to avoid the robot terminating the current instruction due to the new Move instruction. Even though the high-speed mode of the UR3 was used to send the instructions through Ethernet, latency was present. So, a strategy for the instructions queue to reduce this latency may be applied.
2) The use of a Robotic Operating System (ROS) to control the robot for printing is a great opportunity and challenging task due to the multitasking control and synchronization of subtasks. As far as it is known, the use of ROS for printing has not been reported.
3) Another issue was that the size and shape of the extruder that does not allow a completely free movement since the extruder collisions with the support when there are pronounced slopes.
4) I think the work goes beyond printing applicability since precise trajectories for the robot can be generated without the need of knowing how to program a robot or using Polyscope (talking about the UR3 programming software). The work presented here may be used for any robot or application that requires conformal

trajectories on tessellated surfaces in an easy-going and straightforward way to achieve the goals.

Thanks so much!

Bibliography

Abdelaziz, O., Luo, M., Jiang, G., & Chen, S. (2019). *Multiple configurations for puncturing robot positioning*. *1*(4), 1–19. https://doi.org/10.5281/zenodo.2581002

Adams, J. J., Duoss, E. B., Malkowski, T. F., Motala, M. J., Ahn, B. Y., Nuzzo, R. G., Bernhard, J. T., & Lewis, J. A. (2011). Conformal printing of electrically small antennas on three-dimensional surfaces. *Advanced Materials*, *23*(11). https://doi.org/10.1002/adma.201003734

Ahlers, D., Wasserfall, F., Hendrich, N., & Zhang, J. (2019). 3D printing of nonplanar layers for smooth surface generation. *IEEE International Conference on Automation Science and Engineering*, *2019-Augus*(September), 1737–1743. https://doi.org/10.1109/COASE.2019.8843116

Alkadi, F., Lee, K. C., Bashiri, A. H., & Choi, J. W. (2020). Conformal additive manufacturing using a direct-print process. *Additive Manufacturing*, *32*. https://doi.org/10.1016/j.addma.2019.100975

Allen, R. J. A., & Trask, R. S. (2015). An experimental demonstration of effective Curved Layer Fused Filament Fabrication utilising a parallel deposition robot. *Additive Manufacturing*, *8*. https://doi.org/10.1016/j.addma.2015.09.001

Aref, H. (1991). Lindenmayer Systems, Fractals, and Plants (Przemyslaw Prusinkiewicz and James Hanan). *SIAM Review*, *33*(2). https://doi.org/10.1137/1033061

Bobrow, J. E. (1985). NC machine tool path generation from CSG part representations. *Computer-Aided Design*, *17*(2). https://doi.org/10.1016/0010-4485(85)90248-9

Broomhead, P., & Edkins, M. (1986). Generating nc data at the machine tool for the manufacture of free-form surfaces. *International Journal of Production Research*, *24*(1). https://doi.org/10.1080/00207548608919706

Chakraborty, D., Aneesh Reddy, B., & Roy Choudhury, A. (2008). Extruder path generation for Curved Layer Fused Deposition Modeling. *CAD Computer Aided Design*, *40*(2). https://doi.org/10.1016/j.cad.2007.10.014

Chen, H., Fuhlbrigge, T., & Li, X. (2009). A review of CAD-based robot path planning for spray painting. *Industrial Robot*, *36*(1). https://doi.org/10.1108/01439910910924666

Chen, H., Xi, N., Sheng, W., Chen, Y., Roche, A., & Danl, J. (2003). A general framework for automatic CAD-guided tool planning for surface manufacturing. *Proceedings -*

IEEE International Conference on Robotics and Automation, *3*. https://doi.org/10.1109/robot.2003.1242132

Chen, L., Chung, M. F., Tian, Y., Joneja, A., & Tang, K. (2019). Variable-depth curved layer fused deposition modeling of thin-shells. *Robotics and Computer-Integrated Manufacturing*, *57*. https://doi.org/10.1016/j.rcim.2018.12.016

Chen, T., & Shi, Z. (2008). A tool path generation strategy for three-axis ball-end milling of free-form surfaces. *Journal of Materials Processing Technology*, *208*(1–3). https://doi.org/10.1016/j.jmatprotec.2007.12.142

Chen, Y. D., Ni, J., & Wu, S. M. (1993). Real-time CNC tool path generation for machining IGES surfaces. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, *115*(4). https://doi.org/10.1115/1.2901793

Choi, B. K., & Jerard, R. B. (1998). Sculptured Surface Machining. In *Sculptured Surface Machining*. https://doi.org/10.1007/978-1-4615-5283-3

Cuan-Urquizo, E., Martínez-Magallanes, M., Crespo-Sánchez, S. E., Gómez-Espinosa, A., Olvera-Silva, O., & Roman-Flores, A. (2019). Additive manufacturing and mechanical properties of lattice-curved structures. *Rapid Prototyping Journal*, *25*(5). https://doi.org/10.1108/RPJ-11-2018-0286

Diegel, O., Singamneni, S., Huang, B., & Gibson, I. (2011). Getting rid of the wires: Curved layer fused deposition modeling in conductive polymer additive manufacturing. *Key Engineering Materials*, *467–469*. https://doi.org/10.4028/www.scientific.net/KEM.467-469.662

Feng, X., & Cui, B. (2021). *Curved-layered material extrusion modeling for thin-walled parts by a 5-axis machine*. *May*. https://doi.org/10.1108/RPJ-11-2020-0272

Gries, D., & Schneider, F. B. (2005). Geometric Fundamentals of Robotics. In *Geometric Fundamentals of Robotics*. https://doi.org/10.1007/b138859

Hu, P., & Tang, K. (2016). Five-axis tool path generation based on machine-dependent potential field. *International Journal of Computer Integrated Manufacturing*, *29*(6), 636–651. https://doi.org/10.1080/0951192X.2015.1068451

Huang, B., Singamneni, S., & Diegel, O. (2008). Construction of a curved layer rapid prototyping system: Integrating mechanical, electronic and software engineering. *15th International Conference on Mechatronics and Machine Vision in Practice,* *M2VIP'08*. https://doi.org/10.1109/MMVIP.2008.4749598

Huang, Y., & Oliver, J. H. (1994). Non-constant parameter NC tool path generation on sculptured surfaces. *The International Journal of Advanced Manufacturing Technology*, *9*(5). https://doi.org/10.1007/BF01781282

Hwang, S., Reyes, E. I., Moon, K. sik, Rumpf, R. C., & Kim, N. S. (2015). Thermo-mechanical Characterization of Metal/Polymer Composite Filaments and Printing Parameter Study for Fused Deposition Modeling in the 3D Printing Process. *Journal of Electronic Materials*, *44*(3), 771–777. https://doi.org/10.1007/s11664-014-3425-6

Jin, M., Gu, X., He, Y., & Wang, Y. (2018). Conformal geometry: Computational algorithms and engineering applications. In *Conformal Geometry: Computational Algorithms and Engineering Applications*. https://doi.org/10.1007/978-3-319-75332-4

Jin, Y., Du, J., He, Y., & Fu, G. (2017). Modeling and process planning for curved layer fused deposition. *International Journal of Advanced Manufacturing Technology*, *91*(1–4), 273–285. https://doi.org/10.1007/s00170-016-9743-5

Jun, C. S., Kim, D. S., & Park, S. (2002). A new curve-based approach to polyhedral machining. *CAD Computer Aided Design*, *34*(5). https://doi.org/10.1016/S0010-4485(01)00110-5

Kuragano, T. (1992). FRESDAM system for design of aesthetically pleasing free-form objects and generation of collision-free tool paths. *Computer-Aided Design*, *24*(11). https://doi.org/10.1016/0010-4485(92)90069-M

Lasemi, A., Xue, D., & Gu, P. (2012). A freeform surface manufacturing approach by integration of inspection and tool path generation. *International Journal of Production Research*, *50*(23). https://doi.org/10.1080/00207543.2011.618148

Lauwers, B., Kiswanto, G., & Kruth, J. P. (2003). Development of a five-axis milling tool path generation algorithm based on faceted models. *CIRP Annals - Manufacturing Technology*, *52*(1). https://doi.org/10.1016/S0007-8506(07)60537-X

Lin, R. S., & Koren, Y. (1996). Efficient tool-path planning for machining free-form surfaces. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, *118*(1). https://doi.org/10.1115/1.2803642

Llewellyn-Jones, T., Allen, R., & Trask, R. (2016). Curved Layer Fused Filament

Fabrication Using Automated Toolpath Generation. *3D Printing and Additive Manufacturing*, *3*(4). https://doi.org/10.1089/3dp.2016.0033

Loney, G. C., & Ozsoy, T. M. (1987). NC machining of free form surfaces. *Computer-Aided Design*, *19*(2). https://doi.org/10.1016/S0010-4485(87)80050-7

McCaw, J. C. S., & Cuan-Urquizo, E. (2018). Curved-Layered Additive Manufacturing of non-planar, parametric lattice structures. *Materials and Design*, *160*. https://doi.org/10.1016/j.matdes.2018.10.024

McCaw, J. C. S., & Cuan-Urquizo, E. (2020). Mechanical characterization of 3D printed, non-planar lattice structures under quasi-static cyclic loading. *Rapid Prototyping Journal*, *26*(4). https://doi.org/10.1108/RPJ-06-2019-0163

Mineo, C., Pierce, S. G., Nicholson, P. I., & Cooper, I. (2017). Introducing a novel mesh following technique for approximation-free robotic tool path trajectories. *Journal of Computational Design and Engineering*, *4*(3). https://doi.org/10.1016/j.jcde.2017.01.002

Mladenović, . M., a novic, . M., hmann, . . (201 . Tool path generation for milling of free form surfaces with feedrate scheduling. *FME Transactions*, *43*(1). https://doi.org/10.5937/fmet1501009M

Mladenovic, G., Milovanovic, M., Tanovic, L., Puzovic, R., Pjevic, M., Popovic, M., & Stojadinovic, S. (2020). The Development of CAD/CAM System for Automatic Manufacturing Technology Design for Part with Free Form Surfaces. In *Lecture Notes in Networks and Systems* (Vol. 90). https://doi.org/10.1007/978-3-030-30853-7_27

Murray, R. M., Li, Z., & Shankar Sastry, S. (2017). A mathematical introduction to robotic manipulation. In *A Mathematical Introduction to Robotic Manipulation*. https://doi.org/10.1201/9781315136370

Patel, Y., Kshattriya, A., Singamneni, S. B., & Choudhury, A. R. (2015). Application of curved layer manufacturing for preservation of randomly located minute critical surface features in rapid prototyping. *Rapid Prototyping Journal*, *21*(6). https://doi.org/10.1108/RPJ-07-2013-0073

Rodriguez-Padilla, C., Cuan-Urquizo, E., Roman-Flores, A., Gordillo, J. L., & Vázquez-Hurtado, C. (2021). Algorithm for the conformal 3d printing on non-planar tessellated surfaces: Applicability in patterns and lattices. *Applied Sciences (Switzerland)*, *11*(16). https://doi.org/10.3390/app11167509

Sarma, R., & Dutta, D. (1997). The geometry and generation of NC tool paths. *Journal of Mechanical Design, Transactions of the ASME*, *119*(2). https://doi.org/10.1115/1.2826244

Shembekar, A. V., Yoon, Y. J., Kanyuck, A., & Gupta, S. K. (2018). *Trajectory Planning for Conformal 3D Printing Using Non-Planar Layers*. *August*. https://doi.org/10.1115/detc2018-85975

Shembekar, A. V., Yoon, Y. J., Kanyuck, A., & Gupta, S. K. (2019). Generating robot trajectories for conformal three-dimensional printing using nonplanar layers. *Journal of Computing and Information Science in Engineering*, *19*(3). https://doi.org/10.1115/1.4043013

Singamneni, S., & Diegel, O. (2010). Curved-layer fused deposition modelling. *Journal for New …*, *8*(2), 95–107. http://reference.sabinet.co.za/sa_epublication_article/newgen_v8_n2_a9

Singamneni, Sarat, Roychoudhury, A., Diegel, O., & Huang, B. (2012). Modeling and evaluation of curved layer fused deposition. *Journal of Materials Processing Technology*, *212*(1), 27–35. https://doi.org/10.1016/j.jmatprotec.2011.08.001

Suresh, K., & Yang, D. C. H. (1994). Constant scallop-height machining of free-form surfaces. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, *116*(2). https://doi.org/10.1115/1.2901938

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Leaning*.

Sutton, R. S., Barto, A. G., & Book, A. B. (1998). *Reinforcement Learning, 1st ed*.

Ucamco. (2014). *The Gerber Format Specification*. *October*.

Xie, F., Bi, D., & Tang, K. (2020). A potential field based multi-axis printing path generation algorithm. *International Journal of Computer Integrated Manufacturing*, *33*(12), 1277–1299. https://doi.org/10.1080/0951192X.2020.1815851

Xie, F., Jing, X., Zhang, C., Chen, S., Bi, D., Li, Z., He, D., & Tang, K. (2022). Volume decomposition for multi-axis support-free and gouging-free printing based on ellipsoidal slicing. *CAD Computer Aided Design*, *143*, 103135. https://doi.org/10.1016/j.cad.2021.103135

Yigit, I. E., & Lazoglu, I. (2019). Helical slicing method for material extrusion-based robotic additive manufacturing. *Progress in Additive Manufacturing*, *4*(3), 225–232. https://doi.org/10.1007/s40964-019-00090-w

Zhao, D., & Guo, W. (2020). Shape and Performance Controlled Advanced Design for Additive Manufacturing: A Review of Slicing and Path Planning. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, *142*(1). https://doi.org/10.1115/1.4045055

Zhao, G., Ma, G., Feng, J., & Xiao, W. (2018). Nonplanar slicing and path generation methods for robotic additive manufacturing. *International Journal of Advanced Manufacturing Technology*, *96*(9–12), 3149–3159. https://doi.org/10.1007/s00170-018-1772-9

Zhao, G., Ma, G., Xiao, W., & Tian, Y. (2019). Feature-based five-axis path planning method for robotic additive manufacturing. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *233*(5), 1412–1424. https://doi.org/10.1177/0954405417752508

Zhao, H., Gu, F., Huang, Q. X., Garcia, J., Chen, Y., Tu, C., Benes, B., Zhang, H., Cohen-Or, D., & Chen, B. (2016). Connected fermat spirals for layered fabrication. *ACM Transactions on Graphics*, *35*(4), 1–10. https://doi.org/10.1145/2897824.2925958

Zhao, J., Zou, Q., Li, L., & Zhou, B. (2015). Tool path planning based on conformal parameterization for meshes. *Chinese Journal of Aeronautics*, *28*(5). https://doi.org/10.1016/j.cja.2015.06.005

Zhou, C., Huang, B., & Fränti, P. (2021). *A review of motion planning algorithms for intelligent robotics*. *DI*, 1–38. http://arxiv.org/abs/2102.02376

Zhou, C., Huang, B., & Fränti, P. (2022). A review of motion planning algorithms for intelligent robots. *Journal of Intelligent Manufacturing*, *33*(2), 387–424. https://doi.org/10.1007/s10845-021-01867-z

Published papers

applied sciences

MDPI

Article

Algorithm for the Conformal 3D Printing on Non-Planar Tessellated Surfaces: Applicability in Patterns and Lattices

Consuelo Rodriguez-Padilla [1], Enrique Cuan-Urquizo [2,3,*,†], Armando Roman-Flores [4,*,†], José L. Gordillo [1] and Carlos Vázquez-Hurtado [1]

[1] Tecnologico de Monterrey, School of Engineering and Sciences, Monterrey 64849, NL, Mexico; A00826707@itesm.mx (C.R.-P.); jlgordillo@tec.mx (J.L.G.); cvazquezh@tec.mx (C.V.-H.)
[2] Tecnologico de Monterrey, School of Engineering and Sciences, Querétaro 76130, QE, Mexico
[3] Laboratorio Nacional de Manufactura Aditiva y Digital (MADIT), Apodaca 66629, NL, Mexico
[4] Tecnologico de Monterrey, School of Engineering and Sciences, Zapopan 45138, JA, Mexico
[*] Correspondence: ecuanurqui@tec.mx (E.C.-U.); armando.roman@tec.mx (A.R.-F.)
[†] These authors are part of the Metamaterials Lab Group at Tecnologico de Monterrey, Mexico.

Abstract: In contrast to the traditional 3D printing process, where material is deposited layer-by-layer on horizontal flat surfaces, conformal 3D printing enables users to create structures on non-planar surfaces for different and innovative applications. Translating a 2D pattern to any arbitrary non-planar surface, such as a tessellated one, is challenging because the available software for printing is limited to planar slicing. The present research outlines an easy-to-use mathematical algorithm to project a printing trajectory as a sequence of points through a vector-defined direction on any triangle-tessellated non-planar surface. The algorithm processes the ordered points of the 2D version of the printing trajectory, the tessellated STL files of the target surface, and the projection direction. It then generates the new trajectory lying on the target surface with the G-code instructions for the printer. As a proof of concept, several examples are presented, including a Hilbert curve and lattices printed on curved surfaces, using a conventional fused filament fabrication machine. The algorithm's effectiveness is further demonstrated by translating a printing trajectory to an analytical surface. The surface is tessellated and fed to the algorithm as an input to compare the results, demonstrating that the error depends on the resolution of the tessellated surface rather than on the algorithm itself.

Keywords: additive manufacturing; non-planar printing trajectory; conformal printing; curved layer printing; trajectory projection; conformal fused filament; lattice materials

check for updates

Citation: Rodriguez-Padilla, C.; Cuan-Urquizo, E.; Roman-Flores, A.; Gordillo, J.L.; Vázquez-Hurtado, C. Algorithm for the Conformal 3D Printing on Non-Planar Tessellated Surfaces: Applicability in Patterns and Lattices. *Appl. Sci.* **2021**, *11*, 7509. https://doi.org/10.3390/app11167509

Academic Editor: Giangiacomo Minak

Received: 24 July 2021
Accepted: 13 August 2021
Published: 16 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

cc BY

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

One of the most used additive manufacturing (AM) technologies is fused filament fabrication (FFF), which emerged as a prototyping process. During FFF, extruded material is deposited on a flat platform, and the 3D component is built by stacking layers that cause functional weakness. A new trend in research aims to deposit extruded material using out-of-plane trajectories, following the shape of curved surfaces rather than planar layer-by-layer printing. Non-planar printing has facilitated the fabrication of complex geometries, improving their mechanical properties.

Several terms are used in the literature to refer to non-planar printing, such as curved layer fused deposition modeling (CLFDM), curved layer fused filament fabrication (CLFFF), and curved layer fused deposition (CLFD). Path planning concepts relevant to these terms overlap in terms of following a non-planar trajectory and depositing fused material onto curved surfaces. Conformal additive manufacturing usually features direct write (DW) technologies, causing a new field related to conformal fabrication using FFF to emerge. The literature also describes the use of conformal fabrication to add fused material to uneven or non-planar surfaces via one or more different materials to refer to non-planar printing.

Adapting a Very Small Size Soccer (VSSS) competition for learning robotics in virtual teaching

Carlos Vázquez-Hurtado
School of Engineering and Sciences
Tecnologico de Monterrey
Monterrey 64849, NL, Mexico.
cvazquezh@tec.mx

Consuelo Rodriguez-Padilla
School of Engineering and Sciences
Tecnologico de Monterrey
Monterrey 64849, NL, Mexico.
ORCID: 0000-0002-4089-2854

2022 IEEE Global Engineering Education Conference (EDUCON) | 978-1-6654-4434-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/EDUCON52537.2022.9766477

Abstract— **The combination of games, competitions around the world, and mobile robots cause motivation in students for learning while acquiring practical skills. However, the global pandemic has forced social and physical distancing measures, while the need for distance learning has made the teaching of robotics difficult. The intention of this work is the adaptation of the Very Small Size Soccer (VSSS) competition to be used as a distance learning strategy of robotics, following most of the rules of the original competition. The proposal consists in using free-available software and libraries for 1) modeling a non-holonomic wheeled mobile robot, 2) designing the graphical appearance, and 3) implementing different intelligent algorithms for computer vision, path planning, and decision-making. The main contribution of this work is the general methodology and development of a complete virtual system based on the VSSS, using only open and free software tools easily reachable by students. The fundamentals of mobile robotics (locomotion, perception, cognition, and navigation) have been successfully covered during the development of the system, while significant learning has been achieved by students.**

Keywords—VSSS, mobile robotics, virtual teaching, higher education, educational innovation

I. INTRODUCTION

The study of robotics is not an easy task and several strategies for teaching need to be implemented to achieve the final goal of significant learning and comprehension. Mobile robotics is a tractable and interesting area for teaching robotics since it may not require expensive equipment and it may be combined with games as an excuse of motivation. Thus, the different competitions around the world every year allow students to put into practice the theory of robotics learned in class. However, the pandemic has made it difficult to have in-person competitions and new teaching strategies have been developed such as the use of simulators and virtual team activities.

According to Rubio et al. [1], the fundamentals of mobile robotics relate to the fields of ***locomotion, perception, cognition,*** and ***navigation***. The locomotion mechanism allows the mobile robot to move through an environment by using, for instance, wheels, legs, or an articulated body. Locomotion is solved through the study of kinematics, dynamics, and control theory. Perception involves the use of sensing devices, such as computer vision. Cognition relates to the level of intelligence of the controller that uses signals from sensors to decide on the next action to achieve the target. Navigation is related to the ability to know the current position for path and motion planning using different algorithms. These extensive topics need to be successfully integrated and learned by students.

An important competition in robotics is the IEEE Very Small Size Soccer (VSSS) where two teams of three autonomous mobile robots each, play against each other. The main goal of the competition is the development and implementation of intelligent algorithms for playing, having as goal to win the game. Strict rules are perfectly defined to be followed both for the building or appearance of the robots and the court and during the execution of the game. The match is usually played in a non-reflective wooden rectangular playground of size 150cm X 130cm. Above the field, there is a camera connected to the main computer, which processes the images and controls the robots.

Baldao et al. [2] explain that the main dissimilarity between the teams is centered on their decision-making system, path planning methods, and control and positioning algorithms. They developed dynamic game strategies for teams composed of five VSSS robots, and successfully incorporated already known algorithms such as univector field navigation for path planning, a combination of Delaunay Triangulation and Gouraud Shading for positioning, and trees for decision making. Using their strategy, they ended up winning the Latin American Robotics Competition (LARC) 2019.

After the review of literature, a classification may be suggested, according to their field of study: 1) computer vision problem [3], [4]; 2) path planning problem (Genetic Algorithms [5], [6], parametric curves [7], [8], potential fields [9]); and 3) strategy problem (Reinforcement Learning [10], Neural Networks [11], among others [12]). In this work, the different fields of study are incorporated in a complete project, as a suggestion for teaching robotics based on the VSSS competition and using only computer and software tools.

II. IMPLEMENTATION OF THE VIRTUAL VSSS

A. The complete system

The block diagram of the complete system, proposed as a virtual VSSS for learning robotics, is shown in Figure 1. The main program (server) is developed using the Processing software (https://processing.org), where the tasks for the graphics, computer vision, and TCP/IP communication are implemented. The graphics programming updates the visual location of the ball and robots changing over time, as real as possible. The computer vision task emulates a camera that acquires, recognizes, and processes the current position of the ball and the position/orientation of each robot, and these data are sent to each team (clients). Each team receives the data which is used for planning its strategy of the game and produces new data which defines the next action to be taken. The behavior of the ball and the robots in the field are defined by the Box2D block (https://box2d.org).

Curriculum Vitae

Ma. Consuelo Rodríguez-Padilla was born on October 19,1971 in Irapuato, Gto. México. She earned the Communications and Electronics Engineering degree from the Universidad Iberoamericana Campus Leon in 1994. She worked as a researcher in the Instituto de Investigaciones Electricas in Cuernavaca, Mor. for two years. She moved to Reynosa looking for better salary opportunities. She obtained the MSc in Electronic Instrumentation Systems in 2001 from the University of Manchester Institute of Science and Technology (UMIST) United Kingdom. She worked in LG Electronics during almost ten years. She started teaching in 2012 in fields related to Electronics and Mechatronics Engineering. She was accepted in the graduate program of Science and Engineering in the *Instituto Tecnológico y de Estudios Superiores de Monterrey*, Monterrey Campus in August 2018. She finished her studies with a major in Robotics and Additive Manufacturing in December 2022.

This document was typed in using Microsoft Word by Ma. Consuelo Rodríguez Padilla