ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

AUSTRIAN INSTITUTE
OF TECHNOLOGY

Department of Computer Science and Engineering - DISI

UNIVERSITY OF BOLOGNA

Towards image-guided additive manufacturing

Vision path planning in wire arc additive manufacturing.

Master Thesis submitted by

Giacomo Cerino

for the fulfillment of the requisites
for the degree of
M.Sc. in Computer Engineering

Supervisor
Prof. Dr. Luigi Di Stefano
Department of Computer Science
and Engineering - DISI

Co-advisor
Dr. Pablo Eugui
High Performance Vision
Systems of the Center for Vision,
Automation & Control

2022-2023

To all the people I love, who have made me happy and who make me happy every day.

"Calendars and clocks exist to measure time, but that signifies little because we all know that an hour can seem as eternity or pass in a flash, according to how we spend it.

Time is life itself, and life resides in the human heart."

Michael Ende, Momo

Abstract

This thesis goal is to design a virtual environment able to simulate the inline 3D visual inspection of a WAAM machine. According to the guidelines of the FitAM project, a two-axis camera robot has been theorized and simulated as an addition to the main plasma torch body. Rhinoceros 3D, the Grasshopper plugin, and Python scripting are the main tools employed. The algorithm developed, starting from a gcode (nc) file, extracts the ordered points at which the machine should move for each instant. Then it computes for every point the ideal position where the following camera should be, From there, it performs research to understand which points are inside the Field Of View of the camera. At this point, an evaluation is done to determine if they are in a good position, if they are properly aligned, and if there's continuity with it as seen in the previous operation. The results are displayed in an intuitive way, highlighting every point with a color related to how well that point has been seen. The results obtained from testing two algorithms showed that for simple shapes, the inspection could be defined as autonomous and of high quality, but for more complex shapes, the results led to unsatisfying and imprecise data.

L'obiettivo di questa tesi è progettare un ambiente virtuale in grado di simulare l'ispezione visiva 3D in tempo reale di una macchina WAAM. Secondo le linee guida del progetto FitAM, è stato teorizzato e simulato un robot con telecamera a due assi da aggiungere al corpo principale della torcia al plasma. Gli strumenti principali utilizzati sono Rhinoceros 3D, il plugin Grasshopper e lo scripting Python. L'algoritmo sviluppato, partendo da un file gcode (nc), estrae i punti ordinati in cui la macchina deve muoversi per ogni istante. Quindi calcola per ogni punto la posizione ideale in cui dovrebbe trovarsi la videocamera 3D collegata alla torcia. A questo punto, viene effettuata una valutazione per determinare se i punti siano in una buona posizione, se siano allineati correttamente e se ci sia continuità con quanto visto durante l'istante precedente. I risultati vengono visualizzati in modo intuitivo, evidenziando ogni punto con un colore relativo al grado di visibilità del punto stesso secondo la telecamera. I risultati ottenuti dalla sperimentazione di due algoritmi hanno mostrato che per forme semplici l'ispezione può essere definita autonoma e di alta qualità, ma per forme più complesse i risultati hanno portato a dati insoddisfacenti e imprecisi.

Acknowledgements

I want to express my gratitude to Prof. Luigi Di Stefano, my supervisor, for his guidance and assistance. I would like to extend my appreciation to the University of Bologna for providing the opportunity for me to complete my master's project at the Austrian Institute of Technology. Furthermore, I would like to thank Pauline Meyer-Heye and Dr. Pablo Eugui for their invaluable support and attitude; they made me feel welcome as a newcomer and helped me shape my experiment methods and critique my findings. I thank all of my colleagues at the High-Performance Vision System Center (AIT), my friends, and our coworkers for the good times we shared in social and office settings. I also want to thank my family and friends for their support and encouragement during my studies.

This master thesis was conducted as part of an Austrian national project FFG-number: 41546361 called FitAM which aims to improve the performance of Wire-Arc Additive Manufacturing in several ways. It aims to enhance the efficiency and accuracy of the Wire Arc Additive Manufacturing (WAM) process by dynamically adapting process parameters and using live 3D data for inline re-planning. It seeks to develop WAM-specific simulation methods to determine process parameters that minimize component distortions. The project also intends to implement optical inline monitoring to deliver real-time weld pool measurements and establish inline process control for autonomous and repeatable WAM processing. Furthermore, it plans to generate individual 3D models of the raw component to improve the milling strategy and reduce milling time. Lastly, it aims to explore the possibilities and limitations of using recycled material for WAM processing, with a focus on onsite material reuse.

Contents

List of Figures

List of Tables

List of Abbreviations

WAM	wire-based additive manufacturing. Used interchangeably with WAAM
WAAM	wire-based arc additive manufacturing
DED	directed energy deposition
TIG	tungsten inert gas
MIG	metal inert gas
CMT	cold metal transfer
PMD	plasma metal deposition
PLC	Programmable Logic Controller
LKR	Leichtmetall-kompetenzzentrum Ranshofen GmbH - partner company
RHP	RHP-Technology GmbH - partner company
SBI	SBI GmbH - partner company
MW	ModuleWorks GmbH - partner company
FOV	Field of View
FDM	Fused Deposition Modeling
CNC	Computer Numeric Controlled
VS	Visual Servoing
PBVS	Position Based Visual Servoing
IBVS	Image Based Visual Servoing

Chapter 1

Introduction

1.1 Motivation and problem statement

In an industrial environment, often the quality inspection of what has been produced relies on visual inspection, which is an effective non-destructive testing method [1, 2]. Thanks to optical inspection it is therefore possible to spot imperfections either on a 2D or 3D surface thanks to various computer vision techniques without damaging the final result.

This project focuses on wire-based additive manufacturing (WAM), which is a subgroup of directed energy deposition (DED). Arc welding processes with tungsten inert gas (TIG) welding, metal inert gas (MIG) welding, or cold metal transfer (CMT) welding, as well as plasma metal deposition (PMD) processes, represent different methods of the DED processes. With low investment and material costs [3] and high material flexibility with good mechanical properties of the manufactured products, the technology is attractive for industrial use [4, 5, 6]. However, the technology is known for its dynamic behavior [7], which makes mastering the process a challenge and is preventing its wide implementation in the industry so far. As it can be seen in figure 1.1 The M3DP machine is massive and is capable of producing voluminous prints.

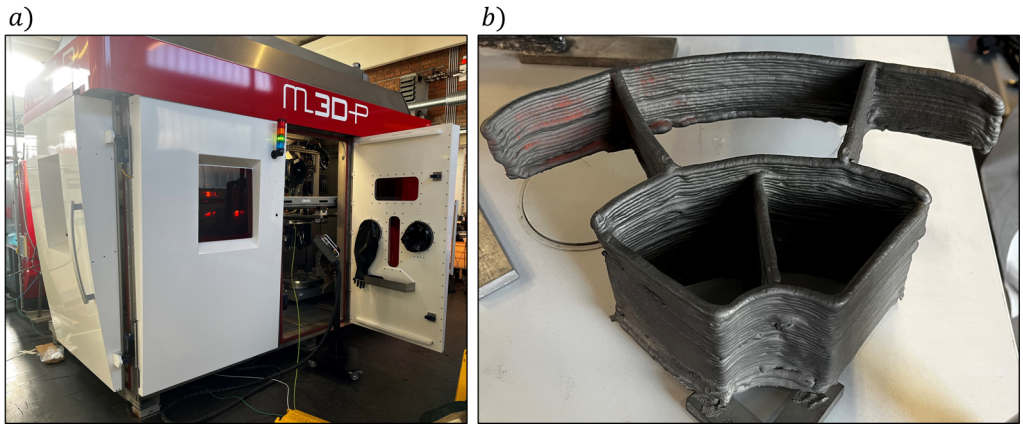

FIGURE 1.1: a) The M3DP machine seen from the outside. b) A titanium part as an example of what kind of products can be made with the machine. This is not the ending product, as it has still to be milled.

The mirror-like, shiny surface of welded titanium is challenging for optical inspection and therefore a good benchmark for optical 3D sensor systems. Despite

the described advantages of the technology compared to traditional machining processes, the unavailability of quality assurance methods currently prevents manufacturers from adopting AM technologies [8].

Nevertheless, the correct choice of process parameters is crucial to avoid distortion and residual stresses, which can lead to material failure and cracking in the part. According to the state of the art, several trials are necessary until the first good part can be produced. A key challenge is the identification of the correct process parameters to maintain geometrical and dimensional accuracy, especially at layer transitions, component corners, and overhangs. Figure 1.2 illustrates the influence of different process parameters on part geometry. 1.2.a) shows an intersecting line geometry welded with standard process settings. 1.2.b) shows the same structure produced with modified process parameters.

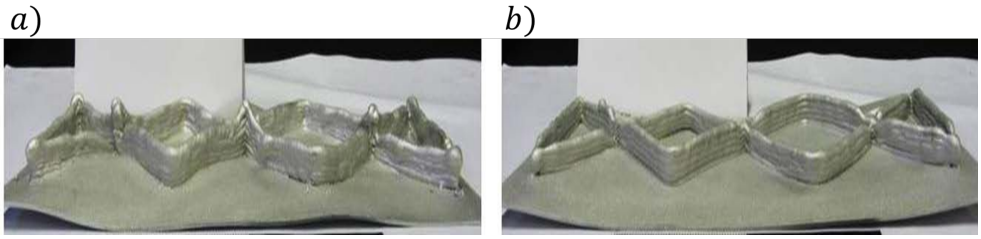

FIGURE 1.2: a) Component manufactured with the standard characteristic curve. The crossing points are higher and overlapped. The metal plate has a wobbly appearance due to the built-up tension caused by the unmanaged spreading of heat during the process, which makes the metal dilate and warp. b) Component manufactured with modified process. As can be seen, the second image looks neater and less warped.

Distortions due to thermally induced stresses, and therefore a loss of geometric accuracy, are a major concern in producing WAM parts [9], especially in the case of titanium alloys [10]. Since WAM is a continuous process in which the filler material is melted and deposited layer-by-layer, the already manufactured geometry is subject to repeated expansion, shrinkage, and temperature changes, resulting in the accumulation of residual stresses [9]. During cooling and unclamping of the welding base plate, the internal stress is redistributed and the component deforms. To mitigate distortions of the final component, it is important to reduce the accumulation of residual stresses during the deposition [11].
Besides the deformation errors, the deposition errors have to be avoided too and this can be done by a reliable quality control system that can spot imperfections rapidly and efficiently. Although in the process of manufacturing ad hoc pieces instead of industrial production, the product should come out without imperfection in the first place rather than being able to spot imperfections afterward. Monitoring the sizes and the geometry of the welded part can prevent unnecessary and wasteful trials from being done by **correcting and adapting** the machining in progress. To achieve this, visual inspection systems are often employed to analyze in real-time the quality of the print and evaluate thanks to image processing techniques the machine's performances in real-time (also called inline inspection [12]). Commonly, inspection processes of this kind focus on the tool's working point. This means that the camera usually looks toward the effector of the machinery. This allows what is going on underneath the torch to be seen, analyzed, and evaluated. However, it's not easy to set up a working inspection system that can work in extreme conditions like the ones

present in WAAM. The torch during the manufacturing process gets so bright that the majority of camera sensors only produce saturated images when looking toward it.

For this reason, special sensors have been developed. For example, Welding cameras can record a very bright environment without completely saturating the image thanks to high dynamic range and light filters. It's important to remember that these cameras even though able to reduce the saturation of the sensor, cannot completely get rid of it because the plasma light and its reflections are still blinding. In literature, these cameras are used to monitor features like the weld pool, which is the portion of metal already deposited but still near the nozzle and therefore still molten. The weld pool is a valuable source of information for example about the temperature in close proximity to the torch [13]. These cameras can also be coupled as 3D stereo cameras to reconstruct the titanium trace as it's being produced. With the obtained 3D model, a comparison with the expected result can be done while the process is still ongoing and can therefore be corrected if needed. Also, laser scanning is a technique that could be used to reconstruct a 3D model of the trace, but the high reflectivity of titanium could cause problems in doing so.

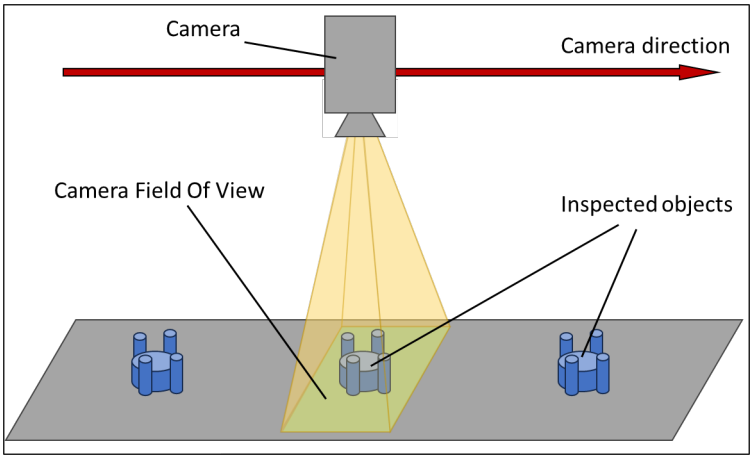

FIGURE 1.3: Basic scheme of the functioning of the ICI algorithm, the camera must mandatorily have a linear and constant movement relative to the inspected objects.

The 3D inspection of the titanium trace opens many possibilities as it allows to understand the height of the deposited layer with respect to the one before and with respect to the expected one, but also the seam thickness and smoothness. An example of 3D inspection technology is, in an ideal case, generating viewpoints to inspect the totality of the product. In [14] an approach to automatically generate viewpoints to inspect an industrial product's totality is presented. However, in this case, several constraints imposed by the project proposition must be taken into consideration, so a less flexible approach has to be adopted. Another technology that has been demonstrated to give great results is the Inline Computing Imaging (ICI) [15, 16], which is an algorithm developed by AIT (Austrian Institute of Technology) capable of using a single sensor to produce a 3D representation of the objects that are conveyed In front of it. This is the algorithm selected for the 3D visual inspection. However, the constraints of ICI are strict as the camera must move along one single direction and with an extremely accurate and predictable motion due to the carefully performed

calibration of the camera. Hence It's crucial that during the movement the Field Of View of the camera is kept perpendicular to the observed plane to keep the measurements reliable. The scheme of the ideal setting for the ICI algorithm implementation is shown in figure 1.3.

For this reason, it is not possible to have the camera looking toward the torch; instead, the camera must perpendicularly look at the welding plate. This is a rather unusual inspection: the camera will not evaluate how the torch is performing in the exact moment that the footage is being recorded, but rather will evaluate what has been printed a few seconds in the past. In this way, directly looking at the plasma light can be avoided, heavily simplifying the 3D reconstruction, and therefore a better representation of the titanium trace can be achieved. Yet some other problems now arise. In figure 1.4 can be seen that until the torch performs prints that follow a simple straight line, the camera can easily be used as an image source for the ICI algorithm. But when the print gets more complex, for example, the printing of a cylinder, the changing in the direction, prevents the camera from looking at the deposited seam so the 3D reconstruction can't be performed.

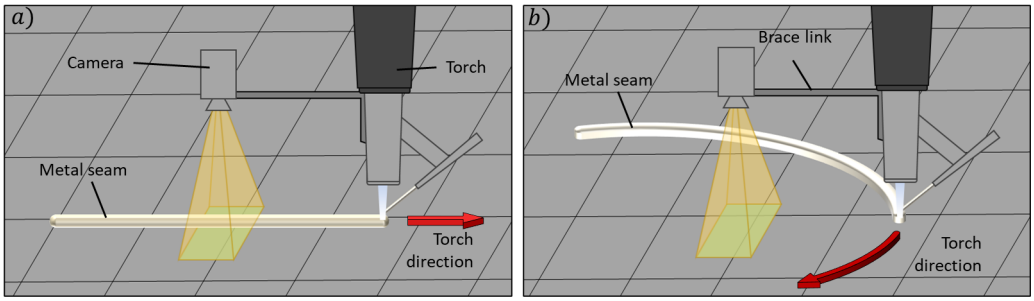

FIGURE 1.4: Scheme of the problem in maintaining the titanium trace inside camera FOV, as can be seen in a) following the trace is not a problem. In b) a more complex path has to be followed and the camera finds itself in the wrong place.

Hence, this research project aims to model a camera moving system and to plan a path investigating if it's possible to inspect complex toolpaths during WAM processes in order to visually and automatically inspect with the ICI algorithm the manufactured piece while it is being produced.

1.2 Contribution/Goal of the thesis

The goal of this project work is to tailor and employ the simulation of a vision system for WAM, enabling real-time virtual monitoring of the welding process while optimizing held of view, scene coverage, and image quality. Ideally, the camera will be fixed to a moving arm capable of orbiting around the plasma torch and orbiting along the camera's vertical axis. This project proposes a simulation framework for machining and imaging processes using computer-aided manufacturing (CAM) software. For this environment to work properly, tools such as Rhinoceros 3D along with the Grasshopper plugin and ModuleWorks' Machining Plugin are used. A path-planning algorithm will be developed to compute the vision add-on

trajectory from a given toolpath which already has several degrees of freedom. A virtual footage can then be produced to simulate the camera's Field Of View. Along with it, also an evaluation software to give a score to the path computed will be presented.

1.3 Structure of the thesis

The thesis articulates in six main parts: An introductory part in which the setting and the problem are briefly presented, a second part in which the state of the art about WAM and visual servoing is explored and explained, a third part reporting the steps performed to build the simulation environment and to develop the path planning algorithm, and the evaluation algorithm. After, a fourth part about the results can be found along with some comments and finally the fifth part is about conclusions and the sixth is about some possible future developments.

1.4 Lexicon

Here follows a list of terms used in the document to refer to specific parts or procedures of this project:

- **Wire or Titanium wire**: the solid wire fed into the plasma torch which melts and produces the seam.
- **Torch**: The tip of the machine's tool, a plasma torch able to melt the Wire and cause the droplets of molten metal to fall on the plate.
- **Seam or Trace**: The welded trace of titanium deposited on the plate or on top of a seam that belongs to the previous layer.
- **Toolpath**: The entire ordered list of coordinates in each of which the torch must be carried
- **Layer**: The portion of toolpath during which one coordinate is fixed, usually the Z axis coordinate, but for complex models the fixed coordinate could be the 4th or the 5th axis
- **Complex models or Complex parts**: Objects produced following the toolpath, which in this case requires also the 4th and 5th axis to be used.
- **Welding plate or Plate**: The metal base on top of which the object will be welded. The 4th and 5th axes of the machine affect it causing its spinning along the central perpendicular axis and its tilting following a fixed direction.
- **Weld pool**: The area of the seam nearest the torch. This zone has yet to fully solidify and therefore is considered liquid and particularly reflective.
- **Camera-axis**: In this report, an axis will be referred to as a camera-axis if it's an additional axis to the five that the default WAAM machine already has.
- **Part or welding**: The final object that has to be produced. ("printed")
- **To Produce or to Weld or to Print**: All verbs refer to laying a trace of molten metal following precise coordinates.

- **.nc file**: it's a file that contains a list of Gcode commands that are to be read by a CNC machine. Depending on the command the machine performs actions such as movement, extrusion, milling, and many others

Chapter 2

State of the art

2.1 WAM technology

As was mentioned in the introduction, the WAM process is a subgroup of the DED processes. In this section of the thesis, it will be briefly explained how this state-of-the-art technology works.

2.1.1 Concept

Additive manufacturing (AM) is the process of joining material to make parts from a 3D model, layer by layer [17]. Another type of additive manufacturing that uses a wire is 3D FDM (Fused Deposition Modeling) printing, where a plastic filament is partially melted and precisely deposited on multiple thin layers [18]. WAM technology operates in a very similar way but mainly differs in the type of material used: in WAM, metal wires are typically used, which can be alloys of titanium, aluminum, etc. while in FDM a wire is used, although not metal but plastic.

In WAM to deposit the material, a wire is used that is melted as it is pushed towards a source of high heat, in this case, a plasma torch. Therefore, the material is deposited below the point of maximum heat of the torch. To achieve the necessary precision, the torch is moved, dropping drops of molten metal onto a welding plate through a CNC numerical control system. It can have multiple degrees of freedom: three degrees of freedom are typically the minimum (for example the machine taken into consideration in [19]) allowing the torch to move along x, y, z linear directions, but additional degrees of freedom A, B are usually added such as a rotation of the welding plate around the center and/or a rotation around the diameter of the same. The standard for this kind of machining tool is indeed five-axis.

2.1.2 Technical features

Figure 2.1 shows a sketch of a PMD and CMT/MIG torch. In both systems, an argon/inert gas arc is used to melt the welding wire and the underlying layer. For the plasma arc of the PMD process, the arc is ignited between a non-consumable tungsten electrode and the workpiece and remains permanently. In the MIG/CMT process, the arc is formed between the wire ("consumable electrode") and the workpiece. In the CMT process, the arc is interrupted, resulting in more smoldering compared to PMD. Employing a gantry system or robot, the torch is moved in the build space under the control of the programmable logic controller (PLC), and a component is built up layer by layer. These processes are particularly characterized by high build-up rates and the possibility to produce large components up to several meters in size [17, 20].

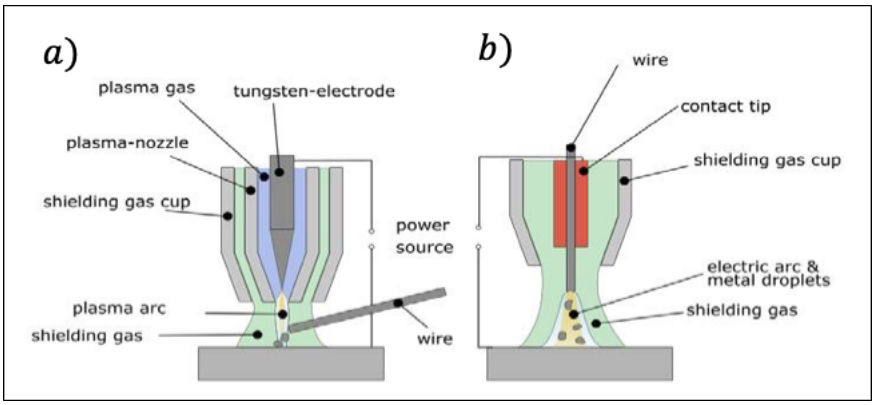

FIGURE 2.1: a) Torch section for Plasma Metal Deposition (PMD) b) Torch section for Metal Inert Gas/Cold Metal Transfer arc (MIG/CMT).

2.2 Vision-Based Control

Vision-Based Control is the name by which the mixing of two techniques that are widely used in robotics can be called. These two techniques are Visual Servoing and Visual Inspection. While the first aims to control the movements of a robot based on visual feedback provided by a vision sensor, the second one is used for quality assurance and it's a non-destructive optical inspection tool that uses vision sensors too. This thesis finds itself in the middle of these two technologies. A Path for a camera must be computed to obtain good footage suitable for Visual Inspection. But also after the simulation, during the machining process, the recorded footage must be analyzed in real-time in order to modify the machine behavior to prevent the print from being a failure and to modify the camera path to keep following the seam. Therefore the state-of-the-art of visual inspection and of visual servoing must be introduced, even though visual servoing is marginal in this thesis as it can be considered the future development of this work.

2.2.1 Visual Inspection

Inspection is the process of determining if a product deviates from a given set of specifications, it usually involves measurements of specific part features such as assembly integrity, surface finish, and dimensions [21]. Therefore it is a quality control task that does not involve direct testing or any contact with the inspected product. The identity of the inspected object is usually assumed and it's not necessary to perform a recognizing process on it. Once an automated inspection pipeline is working at full pace, it allows a 100% checking of the produced pieces, allowing to locate and discard only specific products instead of entire batches as it is usually done with batch inspection techniques. Although automated inspection might seem to be an ideal solution for improving quality and reducing costs, it may not always be feasible. It must run in real-time and be consistent, reliable, robust, and cost-effective. The development cost usually cannot be amortized over many systems, either because special illumination, image analysis, and part orientation restrictions are usually necessary steps in achieving robust system performance [22]. This makes it necessary for the development process to be afresh for each application. Unfortunately not always basic methods such as back-lighting or other lighting constraints

are useful for complex shapes. Furthermore, for automated inspections to compete with the flexibility of human inspectors, the inspection procedure must be able to handle objects that are in unexpected positions and orientations. For this reason, the Automation of the expert inspective and adaptive capabilities of a human machinist would be quite a challenging task for a vision researcher.

2.2.2 Intelligent Inspection programming

Yang et al. in [23] propose an Intelligent inspection planning technique that articulates in three phases: the first one is the study of objects to extract 2D and 3D features and select which ones are the most important ones to inspect to ensure the quality of the product. Once this has been done, they try to find which camera locations and viewing directions should be used to perform the inspection, and in the last part, they research how can the search of the decided entities, and their inspection be robustly and efficiently done once the sensor data is obtained. Once a solution that can observe all the measurable entities has been found, a linear programming minimization is explored to optimize the process and reduce the amount of sensors utilized.

2.2.3 ICI algorithm

Inline Computational Imaging, developed by AIT, is a simultaneous 2D and 3D inline inspection technology which instead of using several small lenses that reduce optical resolution, uses only one multi-linescan camera [15, 16, 24]. The algorithm makes use of two technologies: **Light Field** and **Photometric Stereo**, which together are mixed to achieve an accurate and robust result. The camera, stationary, captures several different images of objects moving inline in front of it. This results in a light-field image stack showing the inspected objects from various observation angles along the transport directions and illuminated from different directions. In figure 2.2.b by looking at two images, it can be seen that the object is slightly moved. The single high-resolution sensor is therefore treated as a stereo camera. The movement of the scene is completely predictable and it's possible in this way to have a high-resolution reconstruction result [25] the light field technique can utilize these stack of images captured from different viewing angles but with the same lighting conditions to effectively determine the height profile of textured areas. Each scan produces 4 stereo reconstructions each with a different illumination, which are later meshed to reduce noise and increase precision. The other technique, photometric stereo, can exploit views of the same objects but under different lighting conditions. The different illumination is obtained by strobing four alternating light sources while the transport stage (visible in 2.2.a) moves under the camera. From these images, shading gradients can be analyzed to compute the surface curvatures of the object and therefore it is an effective approach for reconstructing fine surface details such as pores, grooves, and scratches.

2.2.4 Visual Servoing

Also known as guidance and control, this technique consists of measuring the relative position of objects thanks to visual feedback used in a closed loop to improve the accuracy and performance of a robot system. The vision systems embedded in the robot systems, although not the only embedded measurement systems, are the most common and most used because they allow the robot in question to perceive

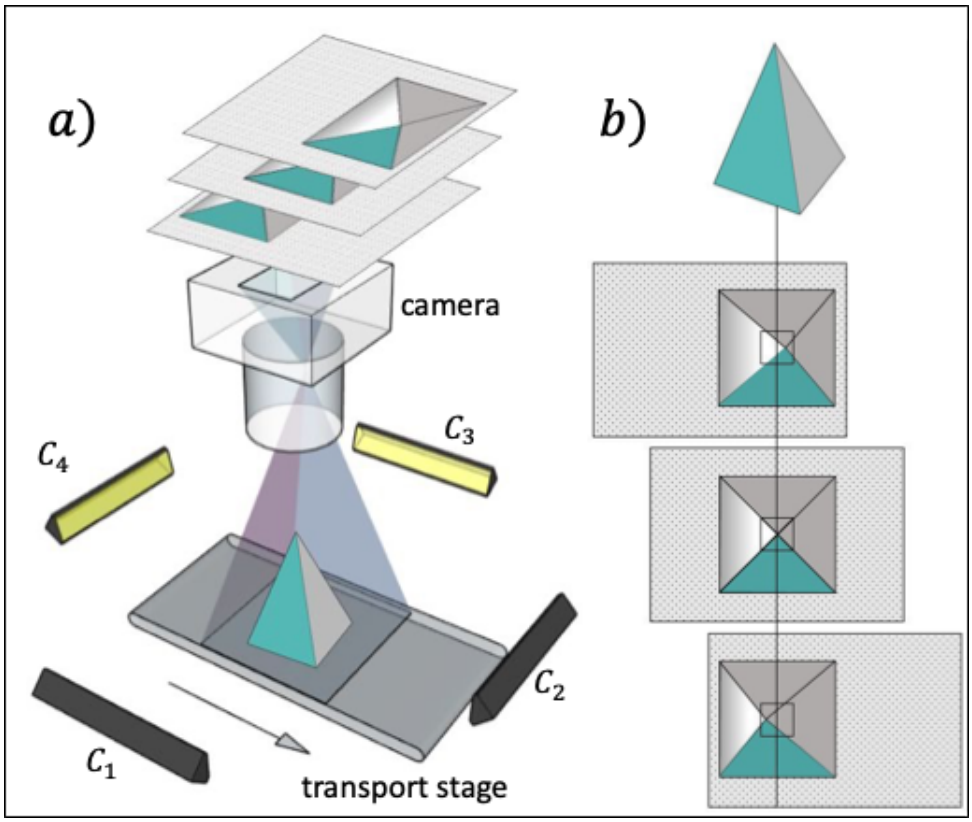

FIGURE 2.2: a) AIT's Inline Computational Imaging (ICI) system consists of an industry-grade camera, four illuminations C1,···, C4 and a transport stage that moves an object (here: a pyramid) in front of the camera, usually at several hundred mm/s. Meanwhile, the illuminations are strobed sequentially, and an image sequence is acquired. b) For each illumination, an image stack consisting of several area scan images is created (only one of four shown here); the transport shift is known and can be used to simplify the finding of corresponding features in the stack. Image from [16].

the surrounding environment in the same way as humans, and therefore allow it to measure the surrounding environment without needing to come into direct contact with it [26]. In literature, there are multiple techniques to do so such as camera in hand and camera on hand as can bee seen in figure 2.3. **Camera in hand** has as a configuration, the camera fixed on the end part of the robot's main tool (called effector) and moves with it, so a transformation between the camera frame and the effector frame must be defined to convert motions from one to another. In the **camera on hand** configuration, one or more cameras are fixed in some points of the environment and can view both the effector and the machined part. In this situation, the transformation between the camera view and the torch position must be computed necessarily for each recorded frame [26].

In general, both problems require a minimization function that tries to eliminate an error

$$e(t) = S(t) - S^* \tag{2.1}$$

Where $S(t) = [S_1(t)S_2(t)...S_k(t)]^T \in R^k$ is the image features at time t; k is the number of image features; $S^* \in Rk$ is the desired image features vector.

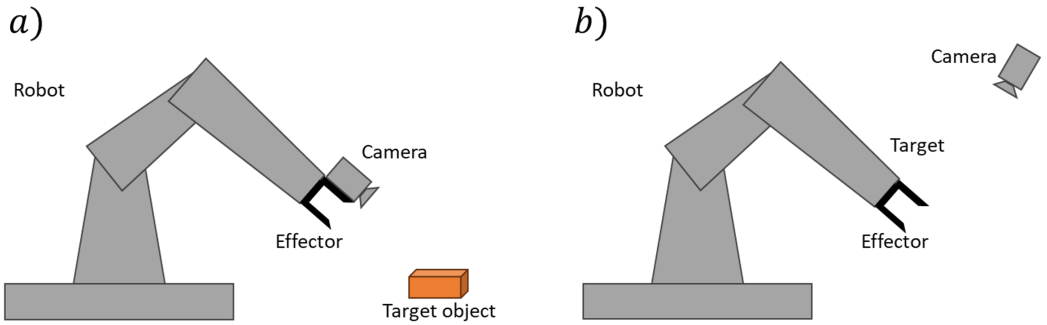

FIGURE 2.3: a) Eye-in-hand configuration: the camera is attached at the end of the effector. b) Eye-on-hand configuration: the camera is fixed and the robot can be seen in his FOV.

Our setting, however, does not use the vision system to move, but rather to inspect and modify internal parameters of the machine not directly related to movement, like torch power and temperature. Indeed, a minimization approach in this case would not be possible, as the system would have to continuously search for different feature points along the titanium track deposited on the plate. In this setting, we are not provided with markers or known feature points, and therefore the planning of the robot's movements will have to be done a priori. Maybe, in future development, a hybrid mode could be developed in which the simulated footage of the camera is used as a reference for real implementation. Allowing in real time to realize if the behavior of the camera is taking an anomalous or unexpected trend by comparing the crafted footage and the live one, and therefore making it possible to progressively decrease the error by implementing a minimization function similar to formula 2.1.

In this thesis, visual servoing is considered marginal but it can be said that all the work is done towards the objective to ease up as much as possible the visual servoing process aimed to autonomously tweak the machine parameters which will be done later.

Chapter 3

Towards image-guided additive manufacturing

3.1 The Machine and Software

For this project, the M3DP, a 5-axis WAAM welding machine from SBI is taken into consideration as the reference device.

3.1.1 Welding machine

The WAAM machine insides can be seen in figure 3.1. It can move its plasma torch along 5 degrees of freedom which are the X, Y, and Z axes of any 3D CNC-guided machine (As a 3D printer). Plus, the welding plate can rotate around its center and tilt in one direction (as it was the flat part of a cylinder halved by its height and laid on the curved part) in a range of [-45, +45] degrees. In the machine, the range is moved to [0,90] with the 45-degree value as the flat-plate coordinate. These last two rotational axes will be called respectively 4th and 5th axes or A and B. There is an additional 6th axis that can rotate the torch and the wire around their central axis to guarantee the optimal melting angle and to avoid any collision.

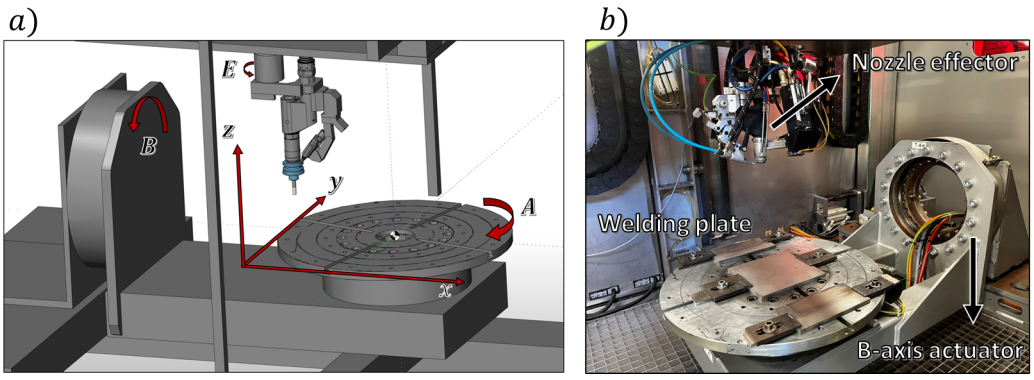

FIGURE 3.1: a) Picture of the machine from the simulation software from ModuleWorks, the 6 axes are labeled with a letter: x,y,z for the linear axis, A for the 4th axis, B for the 5° axis, and E for the 6°. b) A photo of the machine's insides, the nozzle, the welding plate, and the B-axis rotational actuator can be seen

The welding machine encloses all its moving parts inside an airtight chamber. During the welding process, two gasses will be used [17]: a shielding gas passing through the external cavity of the nozzle preventing the molten metal from oxidating by coming in contact with the oxygen. The second gas is an inert gas, like argon,

which will flow in the internal cavity of the nozzle and interact with the arc generated between the tungsten electrode (anode) and the plate (cathode) will generate the plasma flow. This torch reaches extremely high temperatures and therefore can melt the titanium alloy wire. It is shown in 3.2.

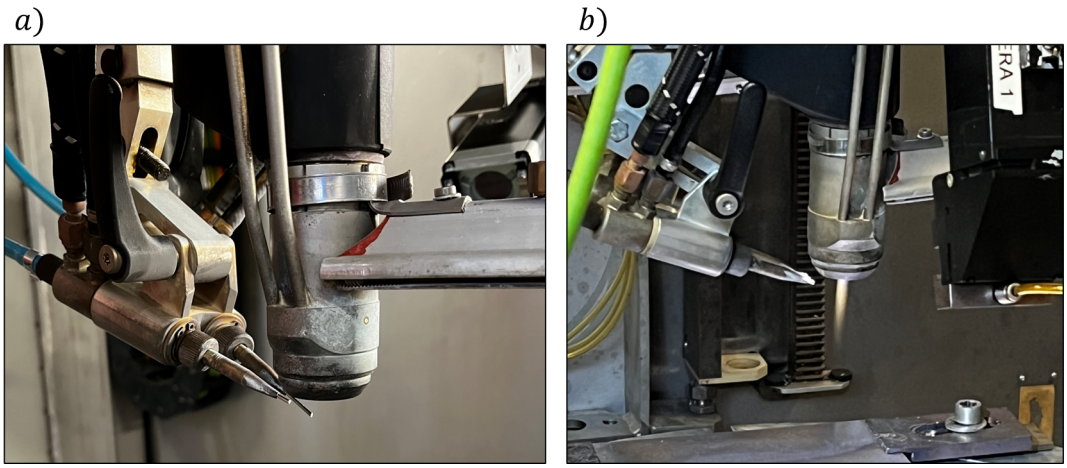

FIGURE 3.2: a) Nozzle of the M3DP machine. The main cylindrical vertical body is the nozzle from which the plasma arc will be generated. It is the torch. Two wires are loaded and can be fed horizontally toward the torch. Usually one is a titanium alloy (Ti6Al4V) and the other one is an aluminum alloy. b) The torch with just the pilot light turned on.

3.1.2 Rhino and Machine builder

The main software designated for simulating the welding process is Rhinoceros 3D, a popular CAD software capable of withstanding, solving, and visualizing complex mathematical problems and equations.

The interface of Rhino consists of four views, which help the user understand how the geometries added to the environment interact with each other in 3-dimensional space. These views can be moved or rotated, and the mode can be switched to display the rendered figure or just the wireframe of the objects, along with many other modes. As a feature, Rhino can automate the movement and the focus point of these views, creating a video of the 3D environment. These automations, however, have to be decided a priori and have to follow a specific guideline. For this thesis, therefore, an additional plugin called Grasshopper will be used to convert this a priori behavior into dynamic and parametrical behavior. This way, the position and direction of a view, which acts in this case as a virtual camera, can be set step by step from the Grasshopper environment. The simulation of the entire machine is possible also thanks to the machining plugin from **ModuleWorks**. Which offers a large number of settings and options to produce the desired toolpath starting from a surface or a solid 3D object. In addition, it offers machine simulation software capable of representing many different 5-axis machines at runtime and visualizing every step of a machining process, also notifying the user with an alert in case of a detected collision.

3.1.3 Grasshopper

Along with Rhinoceros3D, the popular plugin Grasshopper has been used. This plugin is extremely powerful and allows the user to place a series of blocks on a canvas in a very intuitive way, allowing them to connect with wires. Each of these blocks is designed to perform a very specific mathematical, geometrical, or vectorial operation. For every change in the canvas, the solver updates the solution and displays it in the Rhino environment, effectively making the whole experience highly dynamic. It's a perfect environment for parametrical problems and for producing animations.

3.1.4 The ICI algorithm involvement

The footage that the camera should be able to record at the end of this project will be processed and analyzed with the ICI algorithm. As explained in 2.2.3 this algorithm relies on strict constraints. The camera should be properly and exactly calibrated with respect to the inspection stage. Parameters like height and position in space should always be known and precisely checked. Moreover, this algorithm is optimized and is very robust for strict inline movement and for surfaces that have a non-extreme level of reflection.

However, for the setting of this project, the camera will hardly ever follow an inline pattern; even though the aim is to generate the simplest possible path. Even for the simplest rotational print e.g. a circle, the camera must change orientation during the printing process, resulting in an additional part of the ICI algorithm to be developed to map back each recorded point in the right position of the world reference frame. The camera will move in space following the torch movements, and other degrees of freedom will be simulated specifically to improve the capabilities of the system and make it able to maintain the titanium trace centered in the Field of View of the camera and the right orientation. Plus, titanium is an extremely reflective material and because of that, standard vision sensors are incapable of producing satisfying results and must be switched with welding cameras. Moreover, as said in the introduction, the seam inspection will be performed not by looking at the effector but by looking at the welded trace. This is done either to satisfy the ICI algorithm requisites (having the sensor to be oriented perpendicularly with respect to the observed plane). but also to perform the inspection where the light of the plasma torch is not too intense and where the titanium seam has already solidified. However, for the mechanical issues that will arise in the real implementation, the actual distance of the camera from the torch must probably be rather large.

3.2 The Problem

3.2.1 Following the plasma torch path

During a welding process, it is needed to verify that the titanium trace is following the characteristics of the digital-designed model such as seam thickness and layer height with minimal deformation tolerance. To verify the quality of the print during the process, a 3D camera will inspect the freshly deposited metal to measure its width and height in order to produce a 3D model of the manufactured piece. To do so, the camera should hover perpendicularly on the seam, following it precisely and inspecting it continuously as it is deposited, and keeping it in the center of the camera view, possibly always oriented in the same relative direction with respect to the camera field of view. As shown in the figure 3.3.

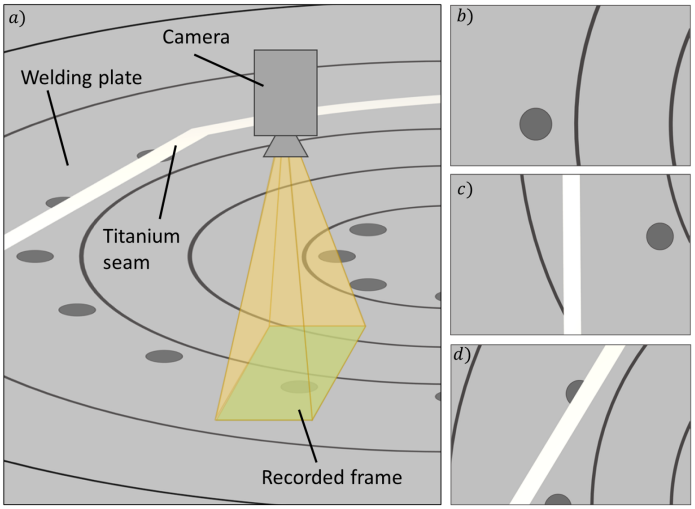

FIGURE 3.3: An example of the welding plate being recorded by the camera. What is contained inside the green rectangle is the Field Of View of the camera. The curved lines and the dark grey circles are the grooves and holes of the welding plate. b) is the frame recorded by the camera in the previous picture. c) An example of a good footage frame. In the background the plate can be seen, in the foreground, going from the bottom to the top, a yellow-gray line can be seen, this represents the titanium trace deposited by the torch. c) A bad footage frame, even though the seam is quite centered, it is not oriented in the right direction.

3.2.2 Collision detection and prevention

The camera will orbit around the torch, but the torch is part of a 5-axis CNC machine that surrounds and encloses the welding plate and torch. So, preventing and avoiding any possible internal collision between the camera and the machine's frame is crucial.

In the setting in which the machine uses all its axes to weld a complex part, the camera should follow the seam. If this task is already challenging for the parts welded with the use of just the X, Y, and Z axes, when the 4th and 5th axes are used too, some new problems arise:

Collision and camera orientation in complex paths

Clearly, by moving the welded piece along the 4th and 5th axis, the possibility of the camera colliding with a previously welded point rises, and the calculation of the possible collision becomes more computationally demanding. The figure 3.4 highlights this problem.

The camera, even though as will be explained will need at least two degrees of freedom, in this situation most definitely will be found in an inconvenient position. Given only two axes to move along, the angle of the camera view will always be not perpendicular to the seam, nor the distance between the camera and the seam will be constant. Figure 3.5 shows the difference between 4-axis prints and 5-axis prints.

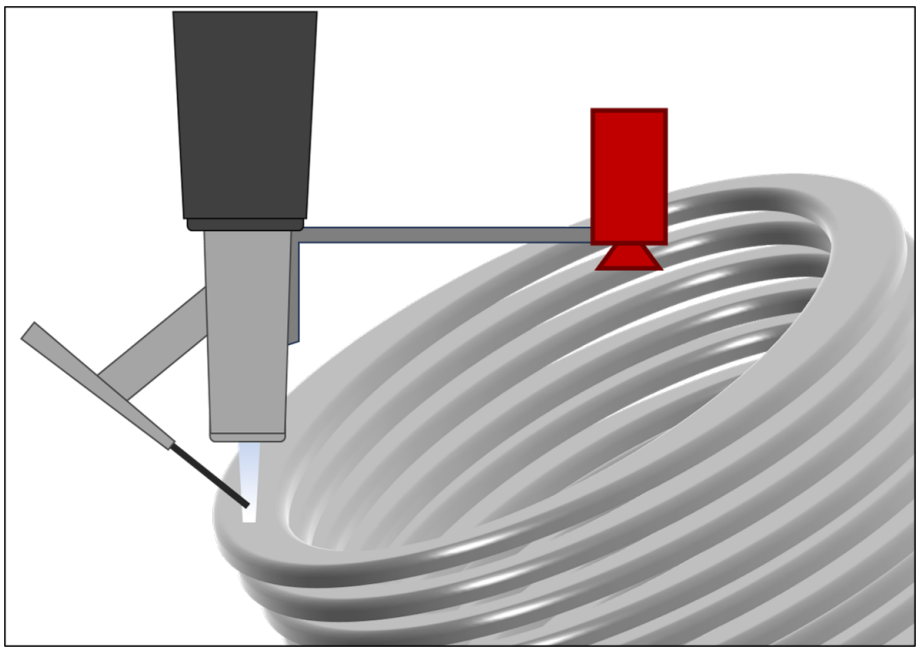

FIGURE 3.4: By using the 5th (B-axis) some problems could arise like the collision of the manufactured part with the camera

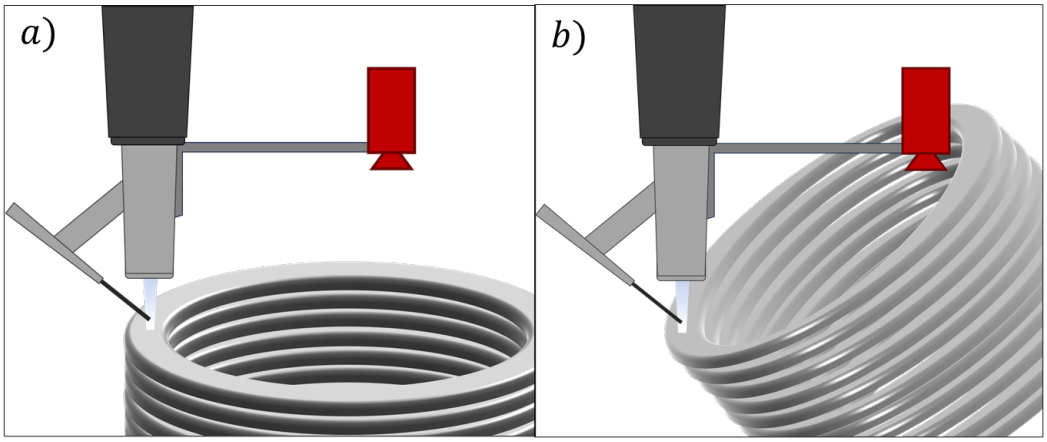

FIGURE 3.5: a) Representation of a part manufactured using only X, Y, Z, and A axis: each layer in this way is parallel to the welding plate. b) Representation of a part manufactured utilizing also the B axis. Each layer could be oriented in different directions and therefore there is a danger of collision.

3.2.3 Constraints

For properly monitoring the seam deposited, is important to maintain it not only in the center of the camera view but also to maintain the overall direction of it with respect to the camera field of view. Therefore, the aim is to obtain footage in which there is a continuous line (the seam) as straight as possible like in figure 3.3.c. This way a good 3D representation can be achieved.

Torch jumps and movements

The torch follows a path. But this path usually cannot be completed without some interruptions: when two lines cross or the metal must be deposited in a place not adjacent to the last point hovered, the torch is forced to stop the wire feeding, lower the torch power, lift the torch and move it to the next good position, rise again the temperature and finally resume the wire feeding. This process is called a jump.

For each jump therefore a large amount of energy goes lost: the carefully balanced torch temperature has to be lowered, and at the right moment it must be increased again using electric power and carefully balanced again. This temperature variation is also time-consuming as these changes are not immediate. Moreover, the sudden loss of heat can compromise the welding result due to the metal shrinkage caused by the temperature variation. Ideally, the torch should never stop during the printing process of each layer. Therefore it's important to keep the number of jumps as low as possible.

At the end of each layer is good to stop for a few seconds if not minutes to give the seam time to cool down and to solidify any melted metal pool left before starting to weld the next layer on top of the previous one.

Number of camera-axis

Hereinafter it will be shown that for sure one single rotational camera-axis is not enough to inspect the quality of the welded seam. So, the problem of how many camera axes are to be considered arises. From the simulation perspective can be added as many camera-axis as needed, but for a practical future implementation is better to keep this number as low as possible for several reasons like mechanical errors and calibration, temperature-induced failure, impossibility of fitting the actuators inside the chamber, and computational complexity. This topic will be further explored but in the meantime, there are some camera-axes worth of notice depicted in 3.6.

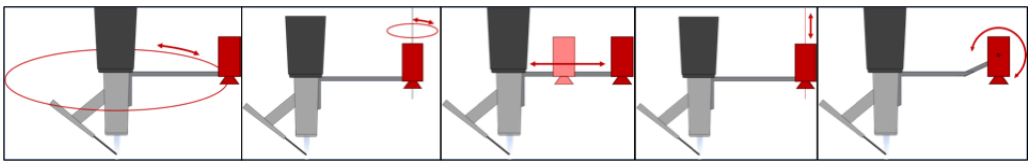

FIGURE 3.6: Image for each hypothetical camera-axis.

a) Torch orbital axis

b) Camera-axis parallel to the 1st (torch orbital) which lets the camera rotate on itself along the y-axis

c) Camera-axis horizontal which moves the camera along the torch orbital radius.

d) Camera-axis to move the camera up and down to increase or decrease the FOV but also for complex models

e) Camera-axis parallel to the plane and perpendicular to the main arm holding the camera. The idea is to tilt the camera in a way that it can look towards the torch or away from it, increasing the range of movement and eventually the percentage of toolpath observable. In this case, however, it must be taken into consideration that ICI works with a very strict movement protocol So an axis of this kind is not feasible.

After evaluating the camera-axis options, only the first two axes have been selected.

Axis constraints and safe area

Another topic to consider is the range of each axis. Ideally, each axis has a 360° range but that is not always feasible, especially in narrow and overcrowded spaces as the welding chamber will be. For this reason, a feature called safe-area has been introduced inside the algorithm which ideally keeps the camera position in a range between [-100,100] degrees from the opposite direction towards which the torch is moving. Figure 3.7 represents an intuitive scheme of it.

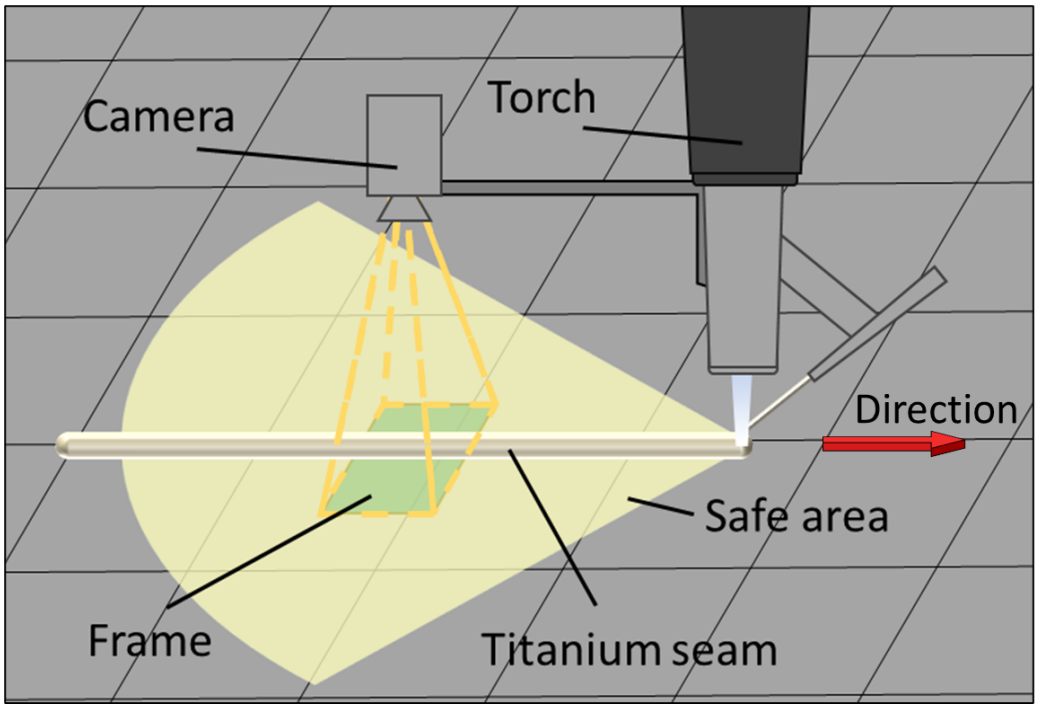

FIGURE 3.7: The safe area represents a portion of the camera domain in which the camera can move and which is computed for each torch position depending on its direction.

Also, for each axis it's important to define a maximum delta angle: the maximum angle which can be walked between two consecutive coordinates depending on how many devices are attached to the torch, the actuator torque power, and the noise

impact on the final footage. Otherwise, some problems can incur such as actuator failure, kinetic shock to the camera, and noise generation or loss of focus.

3.2.4 Using 4th axis

Initially, as a possibility, we thought about using the 4th axis (A-axis), the one which makes the welding plate spin, as a help to have the seam in a good position for the camera in an arrangement where the camera would have been attached to the torch without the ability to move.

With this technique, only a very restricted set of models would have been feasible to manufacture, and more precisely only the cylindrical ones. An example of cylindrical welding can be observed in figure 3.8. As in those weldings, the majority of the movement is performed along the 4th axis.

This idea was discarded very early on, after thinking of having a moving camera. Having the camera capable of moving along some axes on its own allows many more points to be inspected without losing the continuity from one to another.

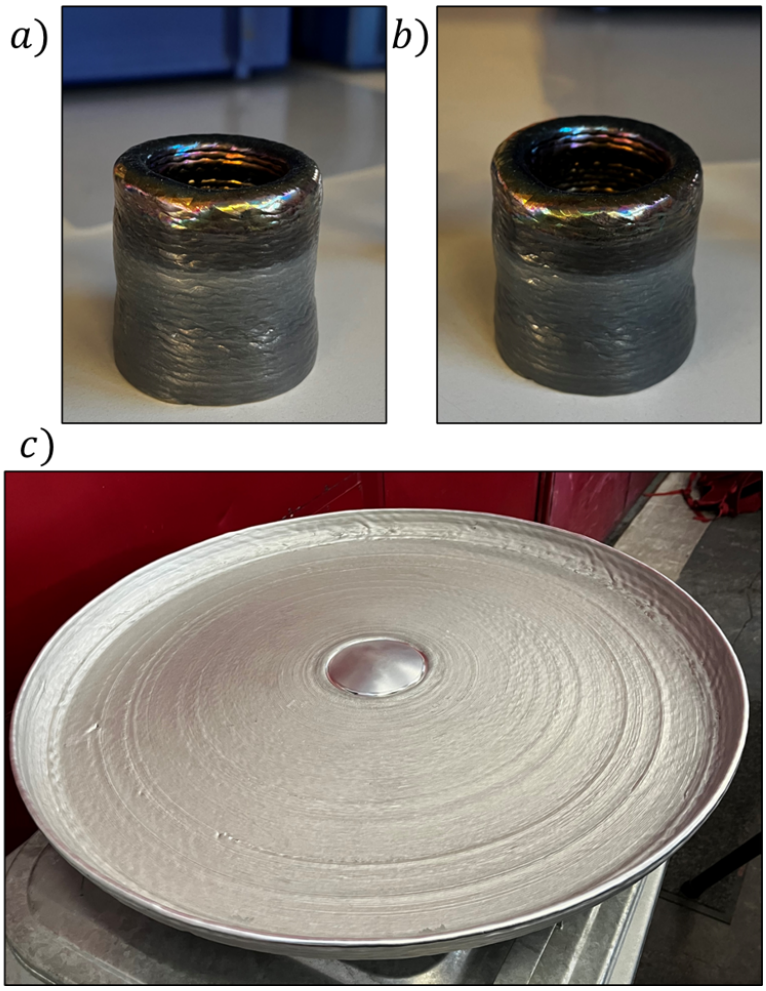

FIGURE 3.8: a),b),c) these products are made through a process called «vase mode» in which the 4° axis (A-axis) is constantly moving and the other axis is sometimes adjusted to grow in height or width.

3.2.5 Online vs. offline approach

One of the first topics we explored was whether we should rather go for a pre-computed offline approach or else for an online real-time approach.

For the latter, the idea was to have an accelerometer sensor mounted on the torch capable of sensing the speed and the direction of it, and simply computing the right vector towards the location the camera had to move to. This calculation could have been a simple inversion of the torch vector or rather a refined value that takes into consideration also the camera constraints.

The alternative offline approach was to compute all the camera coordinates through an algorithm capable of taking into consideration every location of the toolpath and computing the best point to put the camera over for each of the points of the toolpath.

The **online** idea works at an infinitesimal level: derivatives always go in the direction of the curve, but for our setting the distance between the torch and the camera is not infinitesimal but rather large (100mm or more). Computing the right position in real-time by only taking into consideration the data provided by an accelerometer could have been too inaccurate. Computing the best coordinate to observe the seam also means that collision prevention must be taken into consideration and that would have been too computationally demanding for a real-time setting.

On the other hand, with the offline approach, the whole welding process can be simulated and the best path for the camera can be computed more accurately. Moreover, it's possible to identify unreachable portions of the toolpath and try to follow an alternative toolpath.

However, the **offline** approach is immutable once is computed and it would be useful as a future development to have an online tuning mechanism able to notice unexpected behaviors of the machine or the actuators and self-correcting on the fly to maintain the optimal Field Of View of the camera effectively merging the online and offline approach. More about this can be found in chapter 6.

3.2.6 Desired Output

Produce Gcode

The final output of this project, starting from a file "file.nc" which contains all the coordinates for welding an object, is to produce a "newfile.nc" containing also the newly calculated coordinates for each camera-axis added. In a future real-world implementation of this project, starting from the newfile.nc, the machine should be able to recognize the additional coordinates and use them to drive the camera in the desired position during the welding process.

Evaluate the feasibility of the real-world realization

However, for the sake of this thesis, a reliable and promising simulation would be enough to satisfy the requirements and to demonstrate to the partners the feasibility of the realization. Therefore a virtual environment must be programmed in order to be able to observe the torch and the seam behavior. Once the seam simulation is working, the camera simulation can be explored and optimized.

Finally, once the camera simulation works and gives a certain output in terms of how many points of the original toolpath are explored and a score for each point depending on features like position, orientation, and continuity (more on this later). These data can be analyzed and evaluated to compute the quality of the toolpath.

Moreover in Rhino exists the possibility of simulating a camera view step by step thanks to a specific Grasshopper plugin named Animation [27]. Thanks to this it will be possible to realize footage from the camera point of view showing which portions of the toolpath are explored in the best way, which ones are instead hard to explore, and which ones are unreachable geometrically.

In the future maybe will be possible to analyze the footage and to try building a 3D reconstruction of the simulation as briefly explained in 6.1.

3.3 Algorithm developed

The large majority of the code for this thesis has been written for the Grasshopper plugin. As aforementioned, this software is extremely powerful and can handle complex 3D calculations maintaining a flowchart scheme easy to follow and adapt. A picture of the Grasshopper flowchart can be seen in figure 3.9

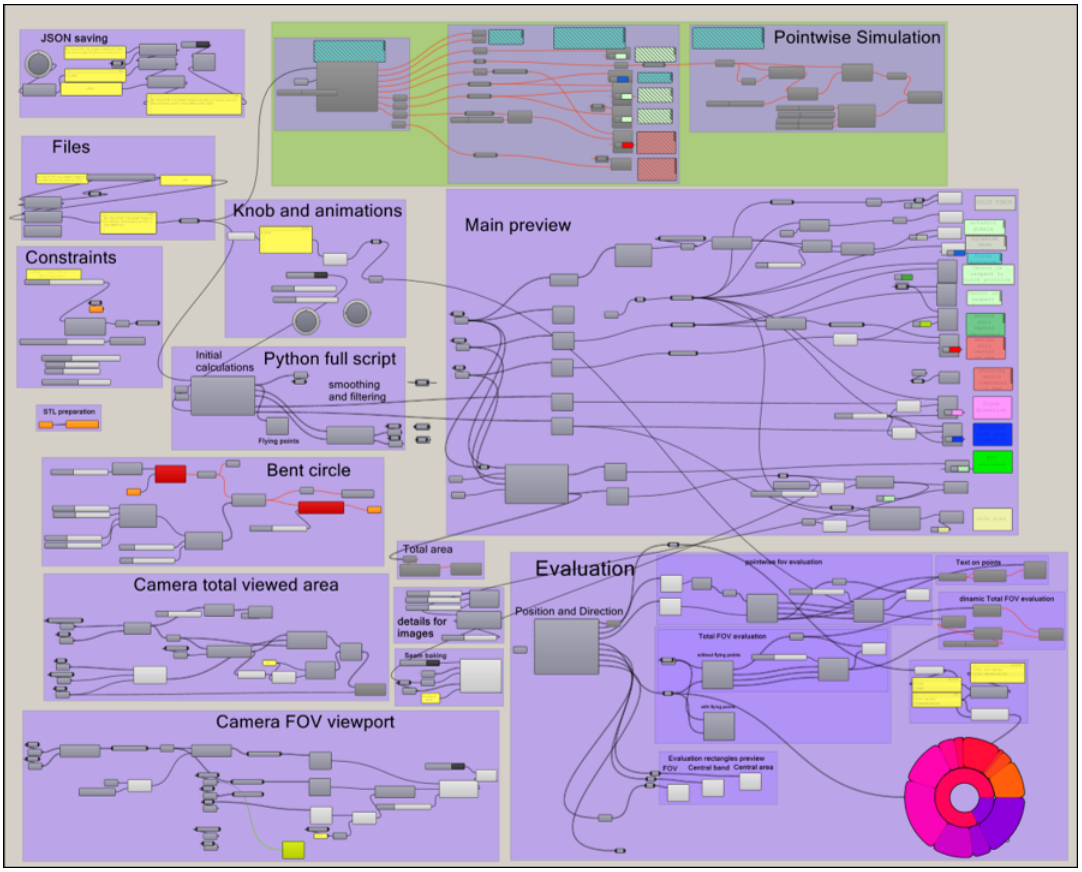

FIGURE 3.9: The Grasshopper script is composed of many blocks, each block receives an input, performs an operation, and exposes one output. These blocks can be divided into groups for better remembering and documenting their functionalities. The output usually is represented as a preview in the Rhino environment e.g. a line, a point, or a surface. To make the preview interactable in Rhino, it is necessary to bake it from Grasshopper. In the bottom right corner a pie chart can be seen, those results are later exported and processed in a separate Python script.

Due to time constraints and skills to be acquired, does not take into consideration some functionalities that are required by the project, but that would have been part of a more advanced analysis. Therefore, it was preferable to simplify the problem. Some examples of simplifications are focusing only on toolpaths with a single layer, adopting a dynamic programming style without having a strict and precise design to follow, making use of theoretical measurements for the camera simulation and position as the bracket and the actuators are still to be designed. These aspects are explored more deeply in 6.

3.3.1 General idea

The general idea for the software to develop is to start from a file.nc in which the welding operation information is stored. An example of these files can be observed in figure 3.10 where several different toolpaths are displayed. By using a Python script as the code for a Grasshopper block, a data structure representing the welding operation will be constructed. This way all the code will be executed inside a Rhino environment and it will be possible later to view the results in real time. Utilizing this data structure will make it possible to analyze each layer of the operation and calculate for each point, the respective camera position. This will be done thanks to Python modules such as Rhinoscriptsyntax [28] or Rhinogeometry [29]. In figure 3.11, 3.16, 3.22, a scheme of the algorithm pipeline can be found, respectively for each section of the algorithm.

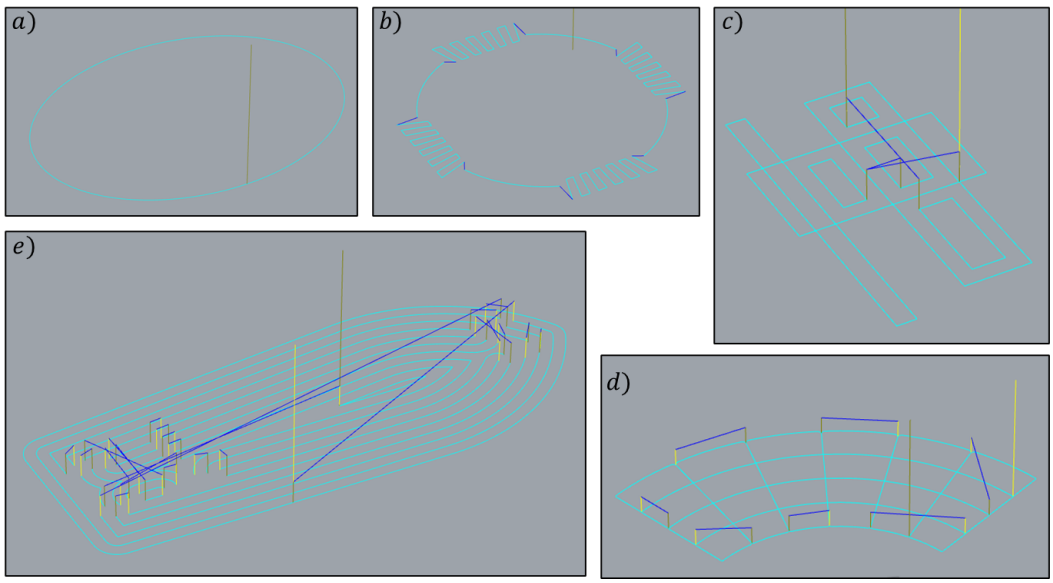

FIGURE 3.10: These several figures display what is contained in a .nc file. a) A Simple path trial. The tool has to move in an elliptical shape. b) An example of a weaving pattern. This is a very difficult stress test for the algorithm because of the impact of the weaving movement on the camera behavior. c) A more complex shape made just by sticking together multiple rectangles. d) A path shaped as a bent grid figure similar to the piece shown in 1.1. e) The first layer of Benchy, a well-known stl file usually used as a benchmark for 3D printers. As can be seen the toolpath in this case is much more complex.

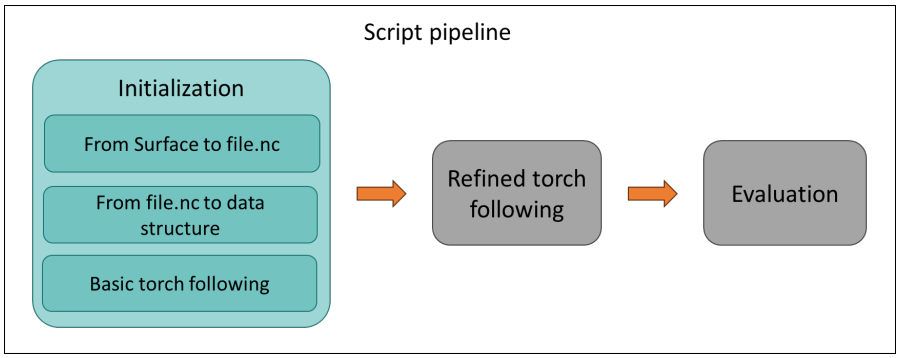

FIGURE 3.11: Scheme of the Grasshopper's script pipeline, the first block, Initialization, is composed of the three steps needed to properly settle the environment and compute the basic path planning.

3.3.2 Process from a surface to a .nc file

The first step of the pipeline is of course to dispose of the .nc file. In the absence of that, a .nc file must be produced with the machining plugin. In a Rhino environment, starting from an object or a surface and one or more guidelines, the Machining plugin will generate a toolpath which later can be turned into a .nc file for a specific machine model of his acquaintance. Toolpaths examples can be observed in 3.10. In this step, several parameters can be tuned to obtain the most suitable and custom machining process.

3.3.3 From .nc file to Grasshopper

Inside Grasshopper is possible to write and run Python scripts. It's easy therefore to read the .nc file and to extract from it the coordinates of each point of each layer in the right order. From this list of points I, all the calculations can be done with a Python script.

3.3.4 Basic torch following script

To follow the torch, first of all, it's important to define what is following the torch and how. A scheme of the system can be found in figure 3.12. The torch will move and a camera will try to follow its same path at a fixed distance. This means the camera will be fixed on a rotating actuator which will allow the camera to turn on itself. Moreover, a bracket will connect the camera and an additional rotating actuator which will let it rotate around the torch on a certain circular domain.

Once the camera distance d has been settled, for each position $i \in I$ in which the torch will have to move, a line l_i is computed by interpolating every position from i_0 to i. This represents the portion of the toolpath already welded. Then a computation is performed to find the points that lay on the intersections between the interpolated line l_i and the circle c_i with a radius equal to d and centered in the position i. This way the points $p_{i,0}, ..., p_{i,k}$ that lay on l and have a distance equal to d from i, the torch position, are identified. The most suitable of them will be chosen by calculating each of the distances on the line l from i to $p_{i,j}$, where $j \in \{0, ..., k\}$ and choosing the one with the shortest distance p_i^* as the "hovering point". A visual representation of the simulation can be found in figure 3.13. In the simulation, the torch is nothing more than three cylinders that hover from one point to another. The green box beside the

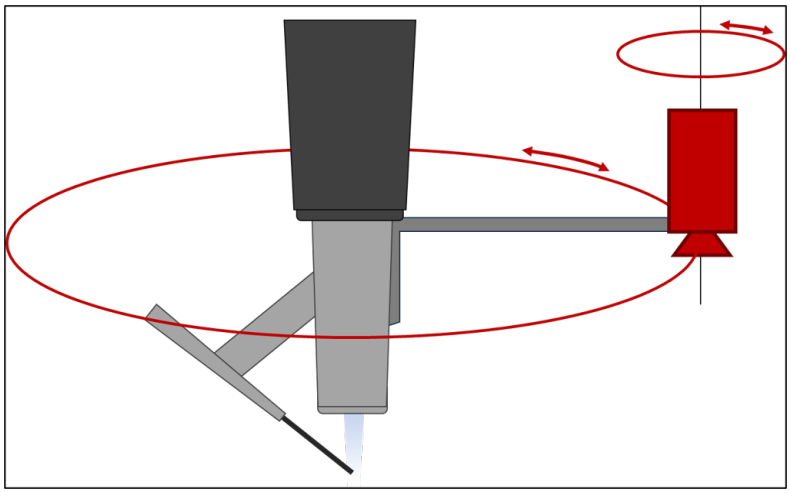

FIGURE 3.12: The torch will have a bracket able to rotate around the torch's vertical axis. At the end of the bracket, the camera will be fixed, and another actuator will allow it to rotate on itself according to the FOV orientation and the situation.

torch is the placeholder of the camera. If the camera is positioned there, the FOV will capture what's inside the green frame on the plane. The light green circle around the blue point is the circle c_i the first camera-axis domain. The camera theoretically can move over every point of this circle.

Once every torch position of the toolpath $i \in I$ has a hovering point p_i^*, a vector v_i from one to another can then be computed resulting in a list of vectors V which contains one vector for each i. These vectors can later be transformed in degrees with respect to a chosen angle, e.g. the starting angle of the actuator. This way the list of angles can be interpreted as the first camera-axis coordinates list.

For each hovering point p_i^* now the derivative with respect to the interpolated line l_i can be computed. The resulting vector is the optimal camera orientation w_i and the list with all the derivatives will be W. The camera must be oriented in such a way as to have a straight "line" going all the way through the vertical central band of the FOV: in this case, the straight line to record is the welded seam. Once W is translated into degrees too, it will serve as the second camera-axis coordinates list.

Simulation insight

At this point, the basic torch-following algorithm is terminated. The script can be contained in a block like the one depicted in figure 3.14. Some data can be brought outside the Python block to better visualize the situation and the performance of the algorithm:

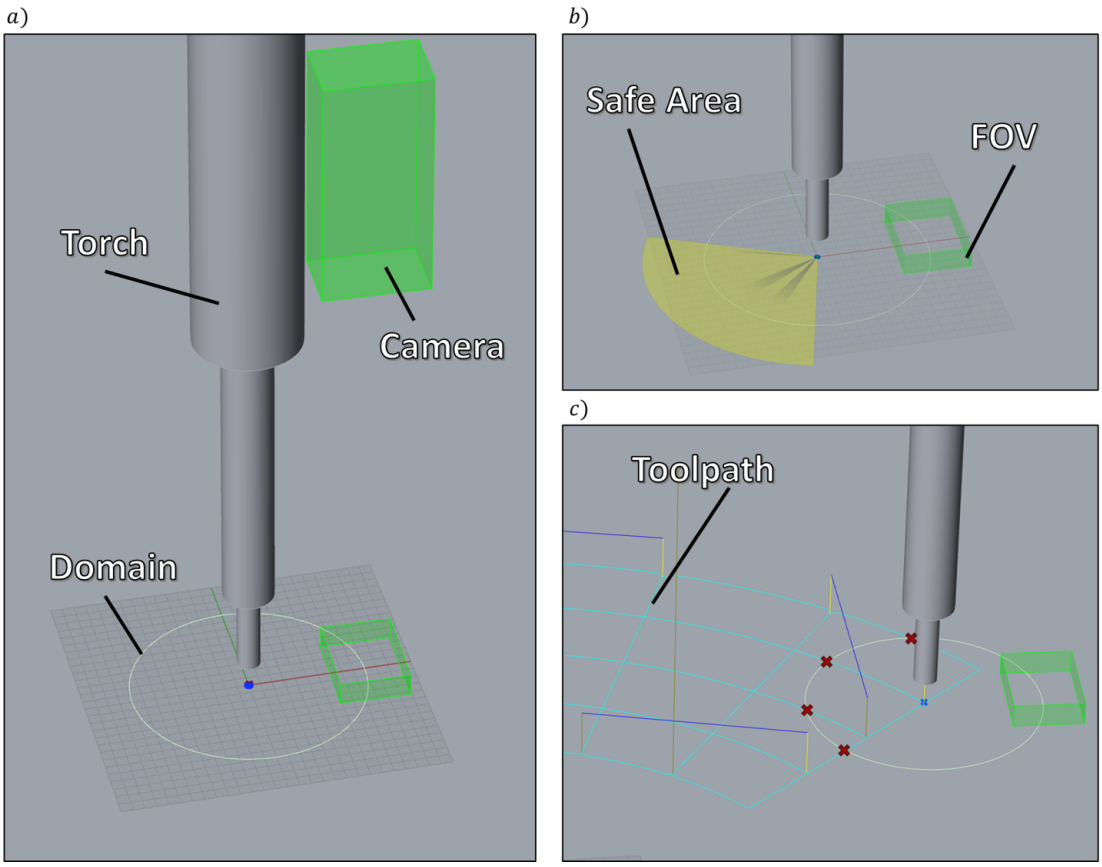

FIGURE 3.13: a) The blue point is the position i on which the torch hovers. i is one of the points of the toolpath I. b) The yellow circle slice represents the safe area inside which the camera should always be kept. This image is a clear example of the FOV being outside the safe area. c) An image of the torch in the starting position of a toolpath. The red crosses are points that intersect the circle c_i, yet they are not valid candidates for the hovering point because the torch has not yet welded any part of the toolpath and therefore the line l_i cannot yet be computed.

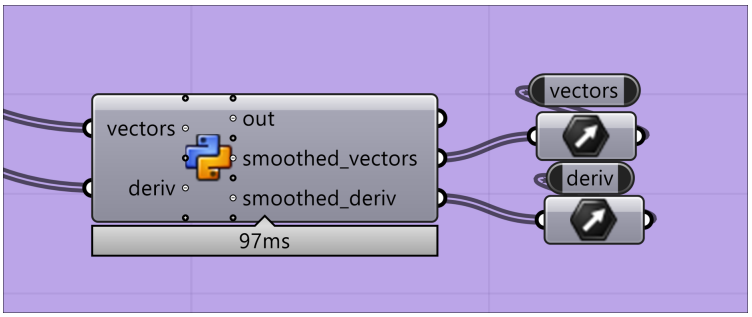

FIGURE 3.14: Example of a Grasshopper's Python block with data coming inside from the left, being processed inside the block, and having data lists coming out from the right part of the block and being stored in vector objects.

The points I and vectors V, W that have been obtained can be used in Grasshopper to build a simulation environment. Precisely the one which can be seen in figure 3.13. In this environment, thanks to a slider, it will be possible to select an index i. i will represent a point of the toolpath I in Rhino and it can also be colored with an arbitrary color e.g. blue in figure 3.13 and 3.15, and by taking the same index from V and W, vectors v_i and w_i can be used to show in Rhino the hovering point p_i^*. As said before v_i is the vector from i to p_i^*, and w_i is the derivative direction otherwise known as the camera orientation. With the help of a few other components, for each of these representations, the position of the camera can be computed, a box placeholder can be placed and its FOV can be shown as a rectangle on the plane. This allows to have a decent view of the behavior of the algorithm and the camera during the printing process.

By sliding the aforementioned slider, in the Rhino environment the shapes will be instantly re-computed and updated generating an animation. An idea of the movement can be obtained by looking at figure 3.15.

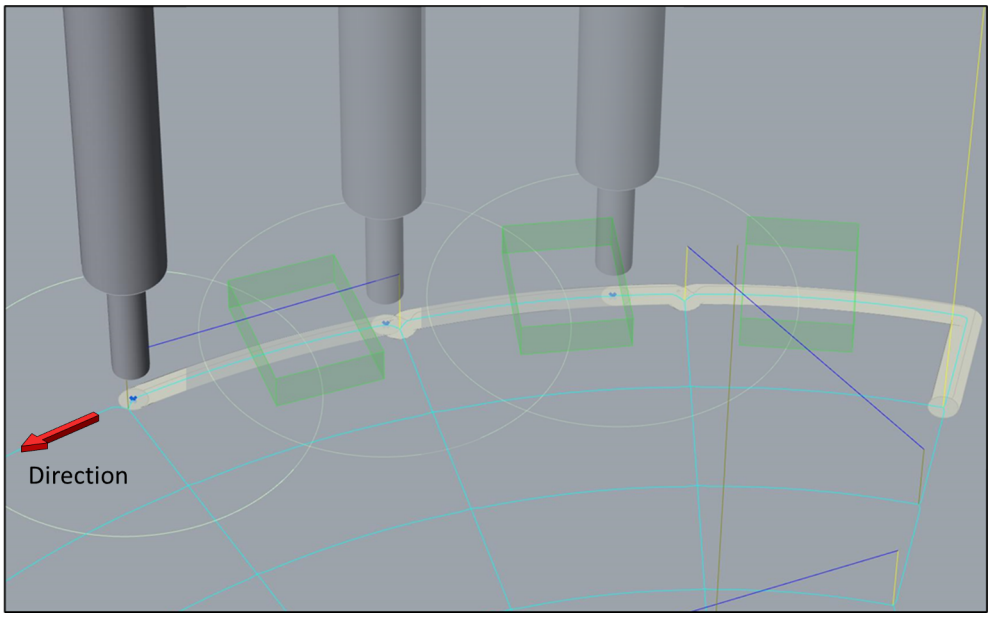

FIGURE 3.15: Representation of the torch moving along the toolpath carrying behind itself the camera FOV (green rectangle) which hovers on a clear trace, the simulation of the welded seam.

3.3.5 Refined torch following script

Once the camera vectors are computed they need some refinements. In fact, by sliding slowly the index slider, an animation can be shown, but this animation is not smooth, but rather rough with frequent lags. Plus, the vectors instead of changing into one another with small and clean movements, change direction frequently and move by wide angles between two consecutive indexes. A scheme of this section of the algorithm can be found in figure 3.16.

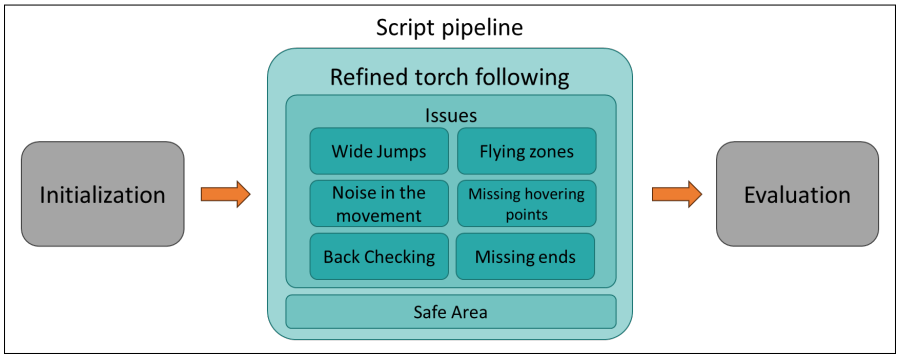

FIGURE 3.16: Scheme of the Grasshopper's script pipeline, the second block is about solving the several kinds of issues encountered. Along with performing the calculations for the Safe Area feature.

Different problems in this section were identified:

- **Wide jumps**: from one index to another, either the 1st or the 2nd axis vector rotates more than a given value which is the maximum delta angle
- **Noise in the movement**: with complex toolpaths that have weaving patterns to fill wide areas with metal, the algorithm finds suitable hovering points. Following one after another these points, however, results in a wobbling movement which is not optimal for the footage recording and for the ICI algorithm.
- **Back checking**: with steep patterns that come from a direction and suddenly perform a U-turn, the algorithm, looking for the nearest point to hover above, finds it in a precedent position on the interpolated line in respect to the previous one. This ends up with several problems during the ICI recording as the algorithm always expects the camera to go toward the same direction and as it introduces information about points already explored.
- **Flying zones**: wherever the torch must lift and move to a point which is not adjacent to the previous one, it must decrease the temperature, to lift and to move to the next point, then it has to restore the operating temperature and finally proceed with the welding. During this traveling time the camera cannot reliably record the deposited seam because of the height difference and because of the free movement of the torch.
- **Missing hovering point**: whenever the algorithm is looking for a suitable hovering point and can't find any, a vector should be found anyway, or else the camera is going to waste precious time putting itself in a more convenient location.
- **Missing ends**: it's impossible to visualize the ending portions of each layer. In fact, when the torch finishes its path, the camera is distant from the last point printed and does any possibility to hover those points

Dealing with flying zones and dark zones

The name that has been given to these zones where the torch is lifted from the plate is flying zones, and an example can be seen in figure 3.17. These zones, in which is not possible to find a suitable hovering point, are called dark zones. A function has

been developed to move the camera during these periods to the next useful position, i.e. the position it will have to take as soon as the torch is back on laying the metal seam. To do so, all the flying zones are turned into dark zones. This algorithm calculates how many instants of dark time there will be and generates the vectors required to move uniformly from the last valid position before the flight to the first valid position after the flight.

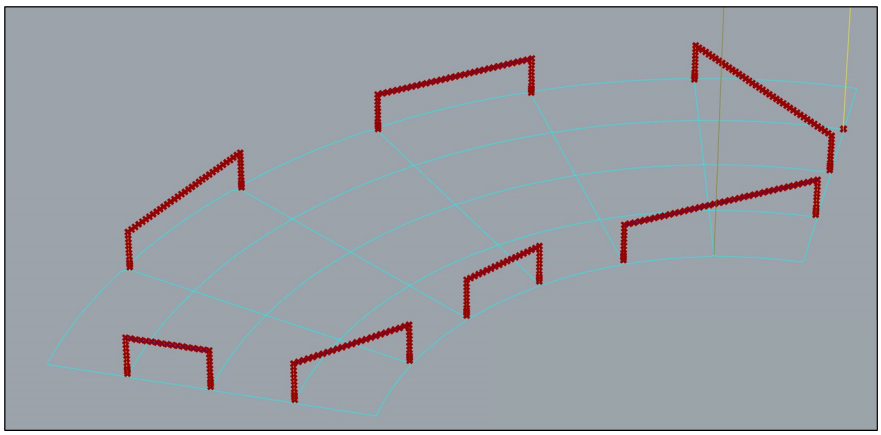

FIGURE 3.17: It is common for non-trivial product processes to have regions of the toolpath in which the torch is lifted and is just moving from one point to another and no material has to be deposited. These regions are highlighted in red. Ideally, the number of these regions should be as low as possible but not always is feasible to print a piece continuously without lifting the torch.

Dealing with noise in the movement

After these corrections to the output of the first script, sudden movements of the camera could still occur, either in terms of angles too wide to be traveled in an instant of time, as shown in figure 3.18 or in terms of the frequency with which these 'jumps' happen.

One idea to reduce this noise was to recalculate each vector by performing a convolution of the list of vectors, i.e. averaging the vectors in their surroundings, this idea was called **vector smoothing**. With this technique, the points explored would not have been right in the center of the FOV frame, but I was hoping to balance these imperfections with the second camera-axis values.

This idea was then discarded as mixing all the vectors by averaging them, as might be expected, would have the effect of changing the path of the camera unpredictably. This is not a desired effect as it is important to maintain the constraint of framing as frequently as possible the exact points of the toolpath in the center of the frame and oriented in the direction of the camera.

Dealing with wide instant jumps

Concerning figure 3.18 and 3.19, the 1st camera axis can only move at one maximum delta angle per instant, therefore a feature called degree-debt (**degdebt**) was introduced to keep track of how many more degrees the axis must move after it has already moved the maximum angle. In this way, in the following instants, the axis

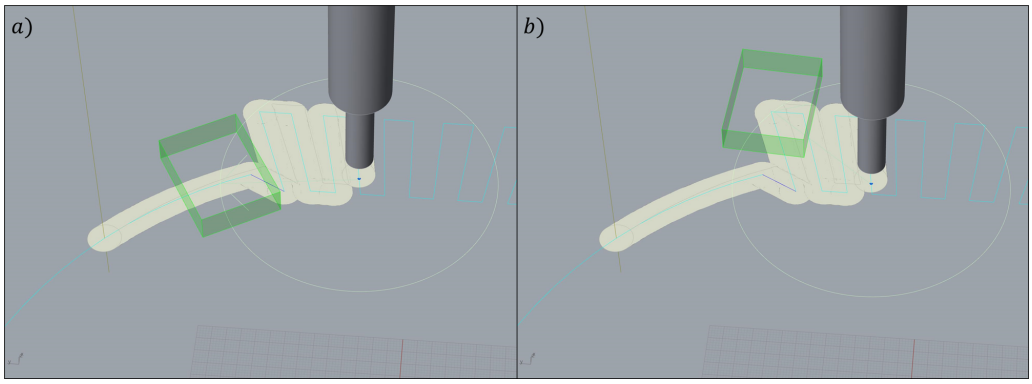

FIGURE 3.18: Two subsequent frames were taken during the simulation on a process with a weaving pattern. This toolpath has two main parts: the first is a curved line, and the second is a weaving pattern. This kind of toolpaths is used to manufacture pieces with wider regions than the average seam width. a) Has the camera FOV on the curved region. b) Is the subsequent point in which the torch has to move, the FOV is far from the precedent point but the movement from one position to another is an angle too wide to be traveled in one instant. This situation would cause a fault or a malfunction during the real manufacturing process.

will move as far as it can to reach the desired angle, automatically increasing or decreasing the degdebt also depending on the new positions it will have to assume in the following instants.

This technique modifies the camera's path but retains the goal of traveling (however more slowly) to the optimal position to observe a point. Whereas with the previous technique, vectors leading to easy-to-inspect points were also modified.

Dealing with back-checking

The problem of **back checking** is very serious and has no simple solutions. Various solutions have been tried to improve the behavior of the camera in these situations. The problem in a nutshell occurs whenever there is a U-turn of the torch path in which the toolpath section from which the torch originates and the section parallel to it, or at least the section following the turn, are less distant from each other than the camera is from the torch. E.g. in figure 3.19 or in figure 3.21. Understandably, this situation occurs extremely frequently, especially since the distance between the camera and the torch is fixed and rather greater (order of decimeters) than the present curves of the toolpath (order of millimeters).

The idea of vector smoothing was also aimed at solving this problem, especially in wave patterns where the curves are very tight and frequent, but it proved to be a failure.

An optimal solution, although costly in terms of calculation and software development, and prone to many risks of collision and damage to the machine, is to divide the toolpath into primitives as it is explained in 6.1.

This and other ideas to prevent and reduce the back-checking problem are explained with a deeper focus on future development 6.

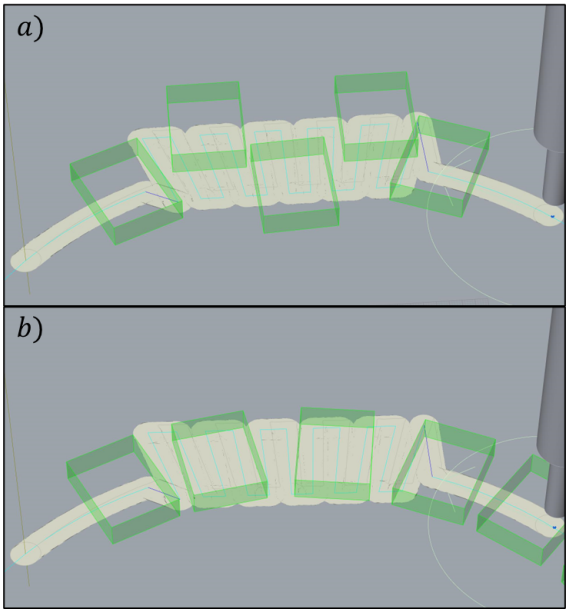

FIGURE 3.19: a) The camera path computed has the FOV wobbling around the weaving portion of the toolpath. This is not ideal as the Inline Video inspection can be compromised by this movement. b) The ideal path would look like this, with a constant and predictable variation in the coordinate of the two rotational camera axes. But to obtain this kind of movement several optimizations are necessary.

Safe Area

Among other ideas to mitigate the problem of back checking, there is one called safe area. The safe area is a region of space that moves with the torch and has the shape of a slice of a circle. As depicted in figure 3.20, the center of the circle is the position of the torch and the radius is slightly greater than the camera distance d. The slice theoretically should be about a hundred degrees wide (a parameter that can be modified at will) and would extend equally to the right and left of the direction of the safe area. The direction of the safe area is nothing more than the direction in which the torch is moving, reversed.

However, the direction of the torch undergoes sudden changes very often in the presence of trajectory changes of any type that are not curved lines, so the list of all the directions of the torch is smoothed out, obtaining milder variations and thus avoiding the safe area between one moment and the next does not have common areas.

The safe area therefore represents the portion of space where the camera is allowed to be in each instant. If the camera is outside the safe area, then the algorithm should try to correct its position in order to bring it back into the allowed space. This back-checking situation occurs when the torch is traversing the part of the toolpath following the U-turn but the camera cannot find new potential points to be explored and therefore returns to hover on points already recorded previously. An example of this situation can be found in figure 3.21.

In this way, it would be possible to avoid a large part of the instants when back-checking occurs by forcing the camera to position itself in an area closer to the next useful position.

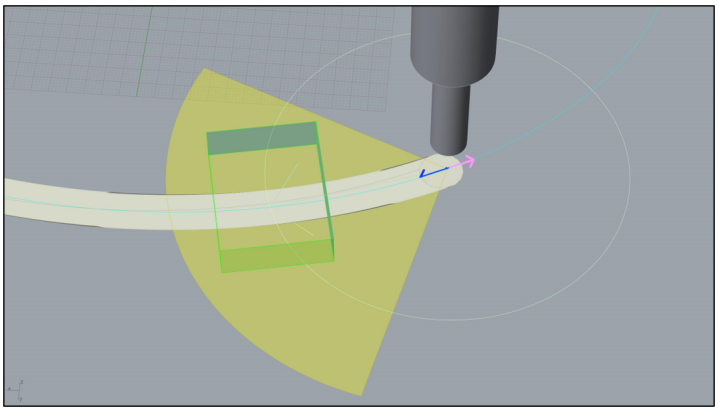

FIGURE 3.20: The yellow circle slice is the safe area. The central point of the FOV must always remain inside this area and an algorithm to comply with this has been developed. The pink arrow represents the torch direction and it is the derivative of the toolpath curve at that point. The blue arrow is the safe area direction, which is not just the same as the pink arrow, reversed, but rather it's an average of the surrounding torch directions in order to obtain a smooth transition between toolpath points.

Animation insight

To follow and inspect the results a whole Grasshopper pipeline has been built which represents the torch position and the camera following it, also providing a visual representation of what the camera is able to see at each moment. To scroll between the moments a control knob can be utilized. A brief explanation of how it works follows.

The Grasshopper pipeline, starting from a .nc file, is able to simulate the torch's path step by step, and for each position of the torch it calculates the position of the camera (first camera axis) and its orientation (second camera axis). With these two pieces of information, it is possible to represent in Rhino the field of view of the camera and observe its behavior in order to verify the accuracy of the algorithm in calculating its path. By leveraging the animation functionality of the Grasshopper slider component, it is possible to generate an image by capturing a screenshot of a specific view of Rhino for each value of the slider and saving these images in a folder on the system. Using video editing software, it is possible to import all the images from a folder as a sequence of frames and use them to generate a video in a chosen format. In this way, the inspection of the camera's behavior is easier to follow and present, and it is easier to notice anomalous behaviors and direction changes, as you do not have to pay attention to the slider on Grasshopper and you also have a better understanding of speed changes during movement along the toolpath.

By taking advantage of this feature with the help of an external Grasshopper plugin called Animation, it is possible, thanks to the added components, to move a Rhino view as if it were the camera, and therefore frame the field of view, managing to produce footage very similar to what will be produced by the real camera in the real setting.

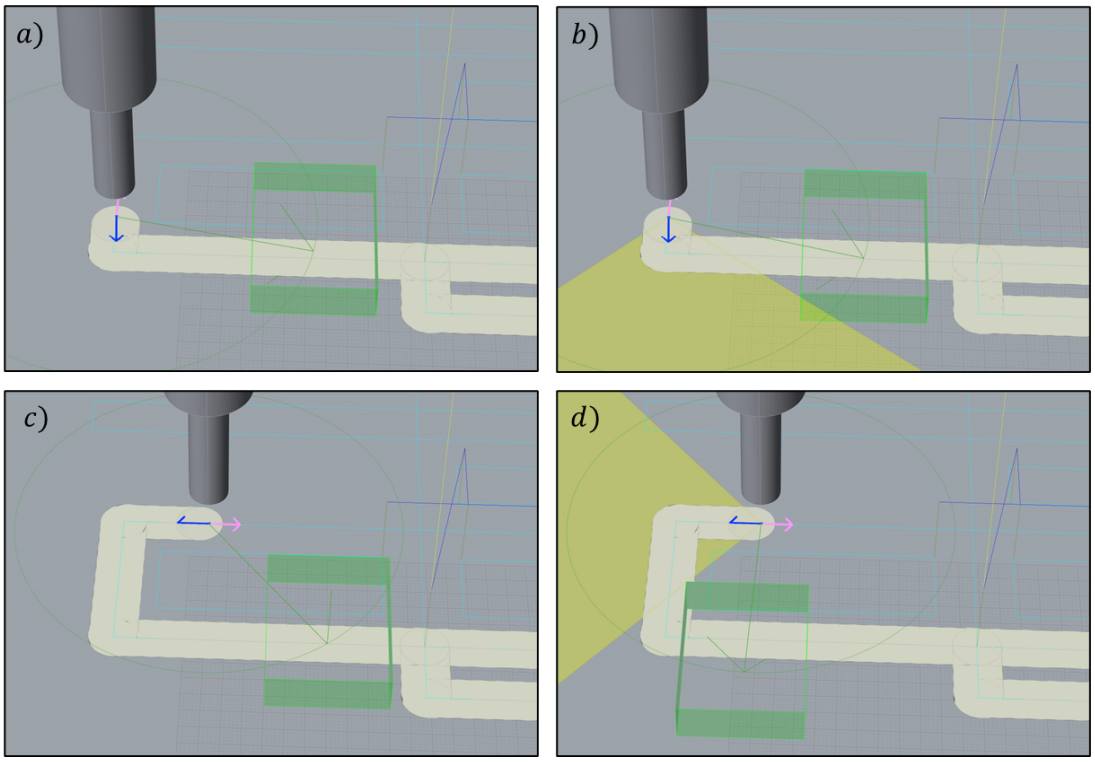

FIGURE 3.21: a) The torch is moving along the pink arrow and the FOV is following it as it is supposed to be, but in c) it can be seen how the torch progressed further in the toolpath and the FOV didn't follow along but rather it stepped back. This is the problem known as back-checking. b) The torch is moving along the pink arrow but this time the safe area option is active. The FOV finds itself outside the safe area so it will try to get back in as soon as possible. In fact in d) can be seen how the FOV is no longer sticking to the toolpath and stepping backward but instead is rotating around the torch to get back inside the safe area again.

3.3.6 Evaluation algorithm

Although it is possible to export the camera's behavior in video format, this is not an efficient way to evaluate the performance of the algorithm. The significant information concerning performance can be listed as the number of points viewed by the camera during the inspection and the quality with which each point is viewed. In figure 3.22 a scheme can be seen of the techniques used to evaluate these parameters.

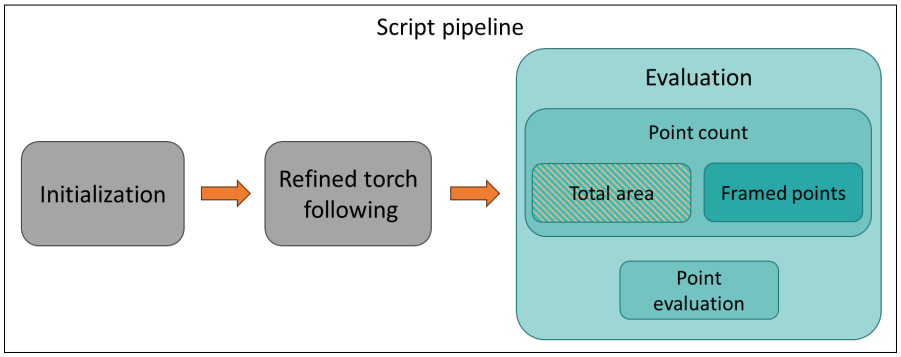

FIGURE 3.22: Scheme of the Grasshopper's script pipeline, the third block is about evaluating the performances of the path planning algorithm. After an initial total area naive approach, a more accurate one has been developed.

Number of points inspected

The toolpath is nothing more than an ordered collection of machine coordinates where the torch must go, combined with some commands that act on the other functionalities of the machine such as the management of the titanium wire feeding or the temperature of the torch. Therefore, initially, just understanding how many of the points where the torch has to go can be inspected is a good measure of evaluation. Ideally, all points should be viewed, so that you can then analyze with a 3D inspection algorithm (for example ICI) the actual quality of the titanium trace deposited and look for imperfections. However, as will be seen, due to geometric constraints, it will not be possible to view all points.

Total area

The first idea was to calculate the **total area** viewed by the camera throughout the inspection, then to overlap the toolpath on the calculated area and detect which points were outside of it, as shown in figure 3.23. The idea, although headed in the right direction, was too computationally demanding, and not scalable to larger prints. And even if some of the points on the toolpath were highlighted as not inspected, the algorithm still had a lot of room for improvement to have a more precise and especially more refined result than just a boolean value (point seen or not seen) and calculated a posteriori. In fact, by performing this calculation, all points of the toolpath are considered, when instead for each instant, the points inside the FOV should be considered valid only if they belong to the portion of the toolpath traveled up to that moment.

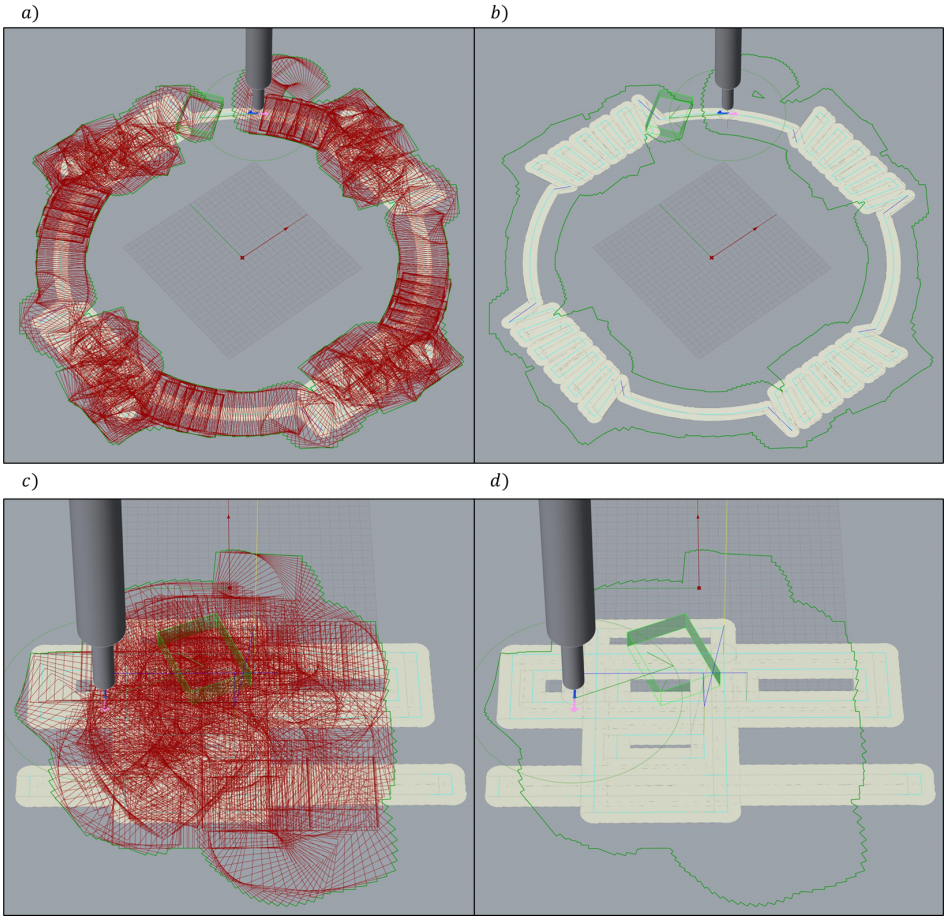

FIGURE 3.23: a) and c) On top of the seam simulation, all the FOV of the camera can be seen at once. b) and d) The union of all the FOV rectangles results in an irregular shape. In these last two images, b) and d), as can be seen, not many parts of the toolpath are outside this complex shape. But the ones which are, are not able to be ever observed throughout the entire inspection process.

Framed points evaluation

So, later instead of the total area approach, I preferred to perform an instant-wise analysis for each calculated Field of View of the camera. The FOV at the simulation level is nothing more than a rectangle drawn in a certain position (position of the torch i translated by the vector v_i which is the first camera-axis coordinate value) and oriented towards the direction w_i which is the second camera-axis coordinate value. Therefore, for each calculated FOV, an analysis was carried out to find out which of the toolpath points traveled up to that moment were contained within the rectangle. Once all the points detected in this way have been collected, it appears which points have been viewed and which have not. This solution is more computationally efficient and produces a more precise result as the points are evaluated taking into consideration only the portion of the toolpath already traveled. An example can be found in figure 3.24.

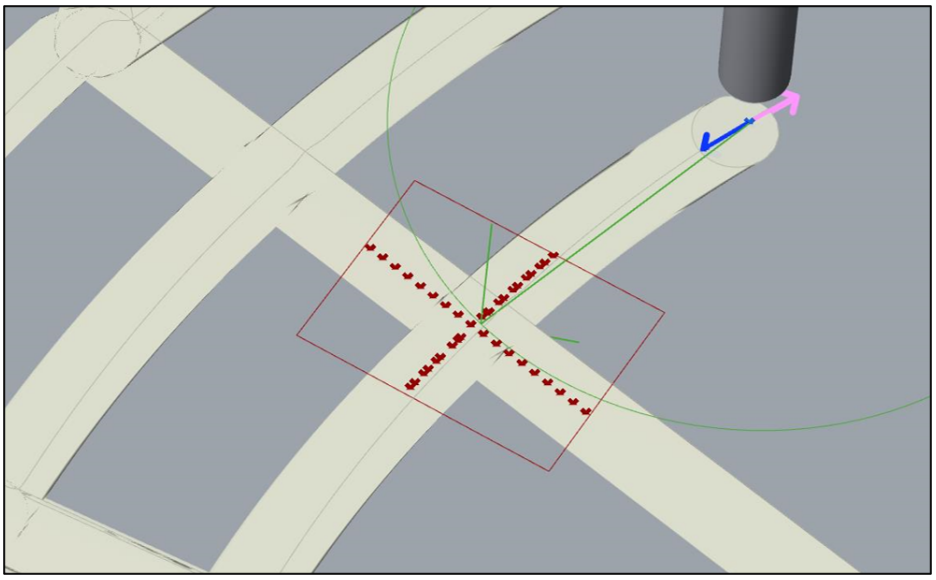

FIGURE 3.24: The FOV of the camera recognizes the points of the path as internal to its borders.

Quality of each point viewed

To improve the quality of the data collected about the inspected points, it is not enough to decide whether a point has been seen or not during the inspection. To perform a 3D reconstruction and analyze it later, the viewed points must be positively evaluated according to the three parameters listed below. According to the importance of each parameter, a score set has been produced. These values are the product of a personal evaluation based on the experience and knowledge of the problem and can be found in tables 3.1, 3.2, and 3.3. While the first table aims to give a strong discrimination between useful points and useless points, the two subsequent parameters aim to preserve the centrality score or to fraction it if the evaluation is not positive.

- Centrality: Within the rectangle that represents the FOV, various areas have been identified. Central area, central band, and peripheral area. As can be seen in figure 3.25. For points to be considered valid, they must be located in the central area of the FOV (green color) or at least in the central band (yellow color). If they are in the peripheral area, they can provide little useful information (red color).

- Orientation: The inspected points must belong to a portion of the toolpath that resembles as much as possible a straight line. To have this type of control, the value w_i concerning the FOV direction, is compared with the derivative δ_j with respect to the toolpath, of each point j which is contained inside the FOV at time i. If w_i, the value of the second camera axis compared with δ_j varies less than a certain value, e.g. 15°, then that point is considered aligned. Otherwise, it is considered not aligned.

- Continuity: A parameter that is extremely important for maintaining the performance of the ICI algorithm, is to ensure that the viewed points are recorded as order-wise as possible, to reconstruct reliably and in real-time the titanium seam. In addition to ensuring that the movement is as linear as possible, the ICI

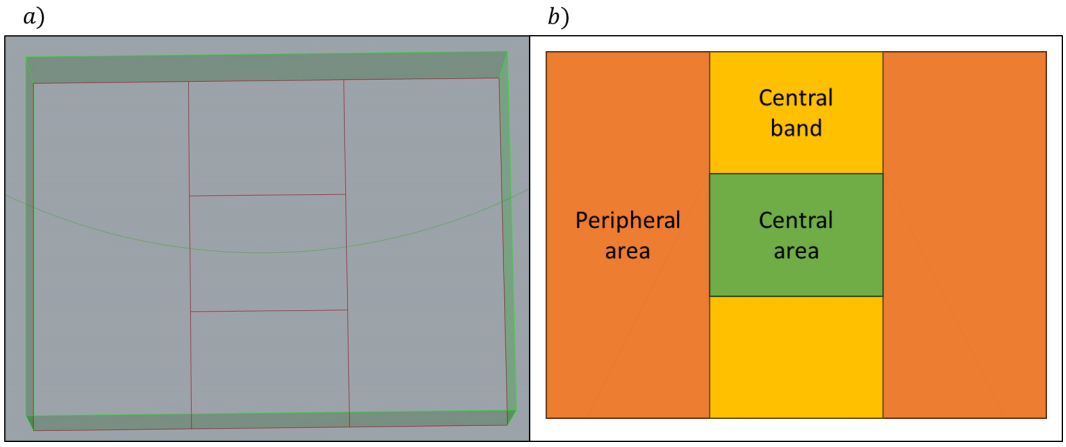

FIGURE 3.25: a) The camera FOV seen from the Rhino view is specifically tailored to mimic the movements of the camera to produce reliable simulated footage of what the camera would see in the real setting. b) Explanation of the regions of the FOV. The central area is the most important one, where the points seen have the best chances of being acquired during the recording. The movement performed by the camera will follow a line, so all the points should appear in the upper part of the central band and disappear in the bottom part of the central band. The peripheral area is not so interesting, many of the points will be seen here but for the 3D reconstruction of the trace, they will not be helpful. In future development, although, those other points could be used to find good points of interest to look for during the inspection in order to synchronize and calibrate the footage.

algorithm predicts that the camera will move "in-line", which means that will follow a linear motion. The continuity is a parameter computed by counting how many times in the past frames, sequential points are recorded together, and in a consistent way i.e. old points disappear and new points appear following the order of the list of points.

In the table 3.4, can be seen how the three characteristics of each detected point are evaluated and mixed through the 3.1 formula to obtain a score. Each point will have a score in the range [-1,4] which identifies it's category. The possible scores that a point can get are 13 in total: 12 visible in table 3.4 which are reduced to 7 possible colors, and an additional one for points that are never seen. A value lower than 0 means that the point has not ever been seen and will be marked with black color in Rhino. A value greater than zero, as it rises towards 4, means that the point has been seen in an increasingly better way and a color progressively more green.

Every time a point is observed, but it has already been observed before. Its previous score is compared with the one just calculated, and only the best score is kept. This takes into account the possibility that a point is viewed for the first time in an unfavorable position and then viewed again later in a better way.

The results of this evaluation can be transformed into a JSON string and saved in the system. Subsequently, thanks to Python modules like Matplotlib and pandas, outside of Grasshopper the data can be appropriately analyzed.

Centrality	Central region	Central band	Peripheral region
Value	4	2	0.1

TABLE 3.1: The position of the point inside the FOV is a foundational parameter to evaluate the performance of the algorithm, therefore this is a good parameter to choose to create a gap between positive scores and negative scores. A point in the Central region is in the ideal position therefore a score of 4 is given to it. The Central Band is in a good position too however the score for this area is just 2. The value is halved because even though is not exactly what are we looking for, a point in the Central band can provide useful information about the continuity and direction of the seam. The Peripheral region instead is not considered a good position so to avoid any contribution a score of 0.1 is given and the gap aforementioned is created.

Orientation	Aligned	Unaligned
Value	1	0.25

TABLE 3.2: This value contributes to the formula by letting the centrality score remain as it is or by dividing it by 4. In fact, a point which is unaligned provides little information as it's not certain that it will provide a good point for the ICI inspection.

Continuity	Continue	Sparse
Value	1	0.33

TABLE 3.3: This value contributes to the point score by reducing it at one third. This is because continuity is considered more important than orientation, even if also orientation is a mandatory requisite for a good point evaluation. Otherwise it leaves the value as it is.

Orientation&Continuity Position	OC TT	OC TF	OC FT	OC FF
Center	4	1.2	1	0.3
Band	2	0.6	0.5	0.15
Periph.	0.1	0.03	0.025	0.0075

TABLE 3.4: This table contains all the possible outcomes of a point evaluation. On the rows, The position is considered and on the columns, all the combinations of Orientation and Continuity (OC) can be found, with a letter T(for true) or F(for false) depending on the evaluation. As can be seen for the useful points green colors are used. For the useless points red color is used, and finally the yellow-orange colors are values that are not ideal, however could be worked on in future developments.

$$\text{Score} = \text{Centrality} * \text{Orientation} * \text{Continuity} \tag{3.1}$$

Simulation insight

As seen in the table 3.4, each score corresponds to a color, which can be attributed within Rhino to the point that possesses that score, resulting in a multi-color view of the toolpath visible in figure 3.26. This makes it possible to distinguish at a glance the portions of the toolpath that can be analyzed the best, analyzed the worst, and those that cannot be analyzed as they are outside the camera's vision range.

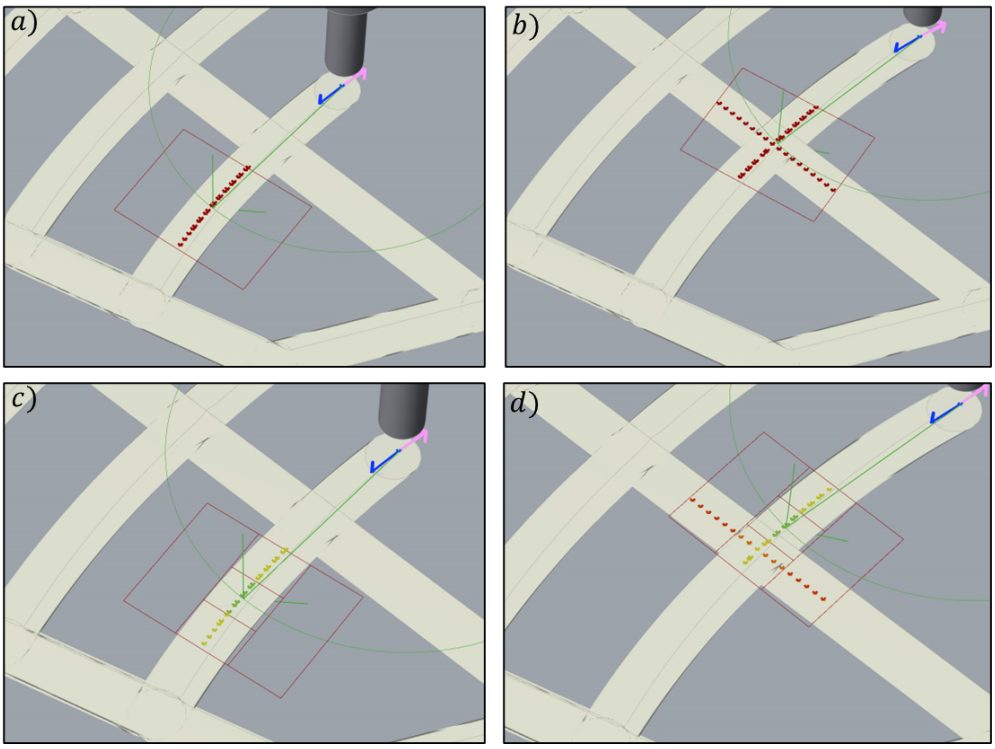

FIGURE 3.26: a) and b) At first, the point recognition functionality has been developed. Later the region-wise and alignment evaluation has been implemented too. c) The points spotted are evaluated because they are all in the center or the central band, oriented towards the right direction, and one after another are all colored in green. d) In this case the points are not all along the central band, some of them are in the peripheral region and therefore they are red, other points are in the central area but in the wrong alignment so their color is orange.

Chapter 4

Results

4.1 Feasibility definition

Shortly after starting to work on the problem and studying it from a mathematical point of view, it immediately became clear that the torch's path could not be followed in its entirety given the constraints set above. Those are, e.g., the fixed distance of the camera from the torch and the limited speed of movement from one point to another in the range of motion. Nevertheless, it was necessary to provide proof of feasibility.

4.1.1 How to measure the feasibility

As stated in 3.3.6, the parameters to evaluate the performance of the algorithm are the **number of points viewed** by the camera during the inspection and the **quality with which each point is viewed**. Therefore, it was necessary to define according to which measure the results would have been satisfactory, as this measure is not specified in the project proposal.

4.1.2 Number of point Evaluation

The best desirable result is a 100% coverage of the observed points compared to those to be observed. Such a result would ensure at least a summary, but global view of the toolpath to be inspected. However, it is rarely possible to reach 100% of the evaluation scale, and therefore it is wise to define percentages according to which to evaluate and collocate the results into different categories.

The manufacturing process is very complex and potentially dangerous if not supervised, so a competent person in the field must always be on-site during manufacturing. Therefore, if the inspected-point coverage reaches 100% the operator will still be on-site, but mainly to keep an eye on other parameters. Instead, with lower coverage, the operator will have to work more to supervise the process during periods when visual feedback is not possible.

In particular, three levels of autonomy have been identified:

- **Autonomous**: This situation occurs when the number of points of the toolpath framed during the entire footage is greater than 95%. In this situation, the percentage of non-visible points of the toolpath is reduced and all in all, it is unlikely for the print to be compromised precisely during the manufacturing of those non-visible points.
- **Semi-Autonomous**: This situation occurs when the coverage percentage varies from 80% to 95%. In these cases, an operator will necessarily have to supervise some portions of the toolpath to ensure correct realization. Despite this, the

manufacturing process is to be considered mostly autonomous and therefore, the inspection method brings considerable help in terms of quality control of the product.

- **Non-Autonomous**: In this situation, is guaranteed a toolpath coverage of less than 80%, which means that an operator will be needed for supervision for a large portion of the manufacturing time. This condition cannot be considered optimal and the goal of facilitating the process, although partially possible, cannot be considered achieved.

Once these three categories have been identified, it is possible to make a preliminary assessment of the feasibility of the visual inspection. This gives a basic idea of the maximum portion of the toolpath that can be evaluated.

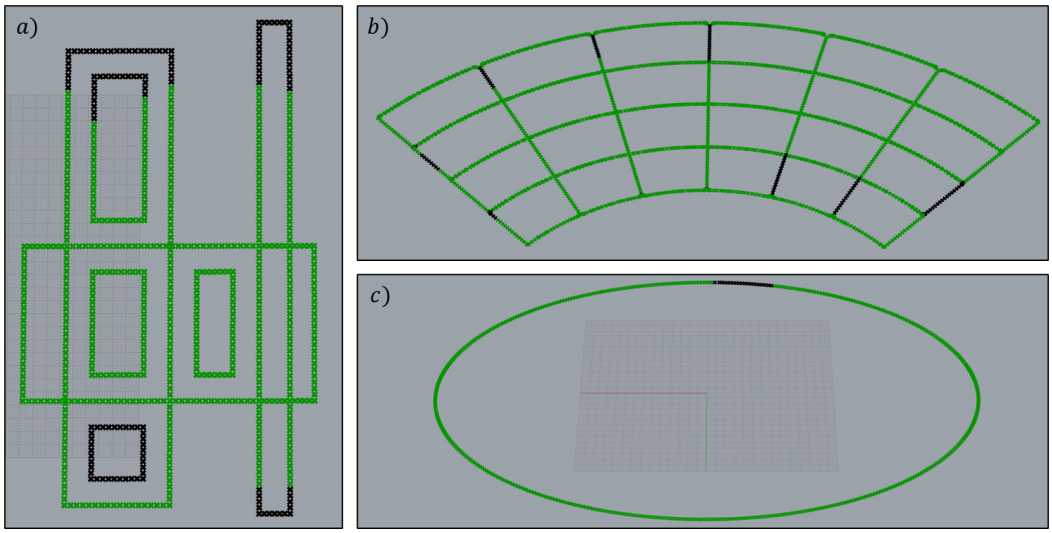

FIGURE 4.1: a), b), and c) show the observable toolpath in green and the not observable toolpath in black.

4.1.3 Quality of point viewed

After evaluating the number of points observable by the camera during the inspection, the second part of the evaluation algorithm is executed, assigning a score to each point depending on its characteristics as explained in the figure [see the figure below continuity]. Also in this case, it is wise to define an evaluation scale for the results obtained, in order to assign a label to the visible portion of the toolpath that expresses the quality of the inspection. Therefore, considering only the viewed points, three categories can be defined here as well:

- **High Quality**: The points inspected during this process are evaluated as aligned, central, and continuous for a portion of the visible toolpath greater than 95%. This contributes positively to the application of the ICI algorithm during the manufacturing process and therefore it is the ideal situation to perform the visual inspection.

- **Sufficient Quality**: The points visible during the inspection are evaluated as aligned and continuous, and recorded in the central band of the FOV, for a portion of the toolpath between 60% and 95%. This means that the evaluated

points are, yes, aligned and continuous but not necessarily central and in any case not for a portion of the toolpath greater than 95%. This will undoubtedly have negative effects on the future implementation of ICI but perhaps with implementations of different algorithms, it will be possible to provide feedback to the machine based on these data.

- **Low Quality**: In this case the quality of the viewable points is low. The camera is probably struggling to follow the toolpath due to too tight and frequent curves and therefore the points viewed during the inspection are largely oriented in wrong directions and recorded within the FOV in non-ideal regions, i.e., in the peripheral area. The number of central, aligned, and continuous points does not exceed 60%. Therefore, in the case of an ICI implementation, it would be very difficult to obtain a useful reconstruction from the footage.

4.2 Results with different techniques

Having also declared these three evaluation criteria, we can now present the results obtained by simulating some toolpaths and evaluating them according to the various techniques listed.

4.2.1 Results with point evaluation

As previously said this technique aims at highlighting every point of the toolpath with a color related to its score, computed with 3.1 and colored according to 3.4. In 4.2 a legend can be found and consulted to better understand the graphs following.

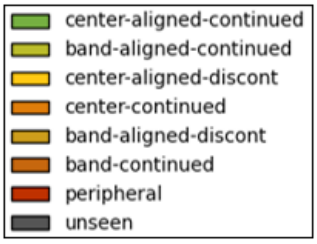

FIGURE 4.2: A legend from the most positive outcome to the worst possible

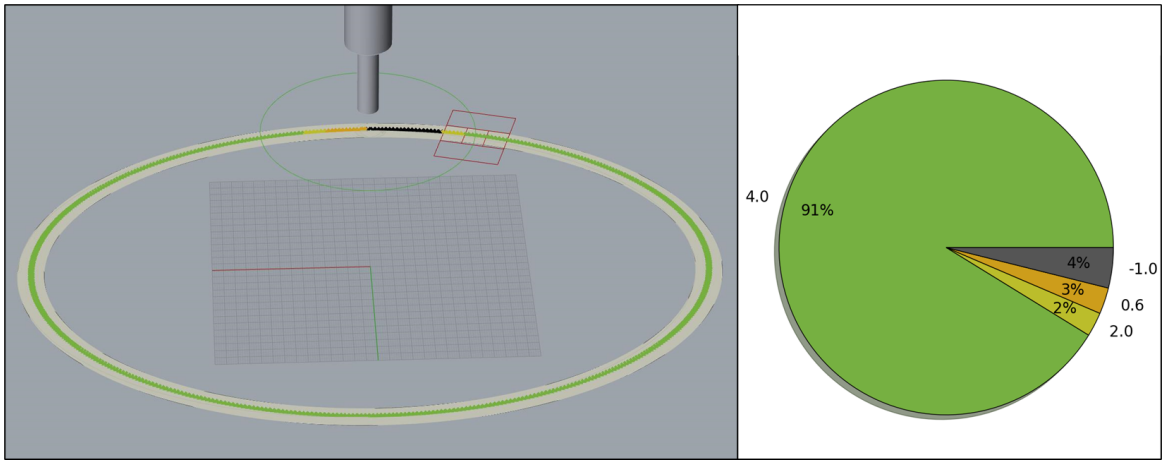

FIGURE 4.3: Quality of the elliptical toolpath with Point Evaluation

In 4.3 the elliptical toolpath result can be observed. The wide majority of points (91%) are rated as center-aligned-continued, which means they are recorded in an

ideal way. this is a good result, but it is also true that this is a very basic shape, without sudden direction changes, corners, or jumps. According to the metric of evaluation defined before, this product is considered **Autonomus** and to have a **High Quality**. In fact, over the totality of the observed points, we get a high result of $93/96 = 97\%$.

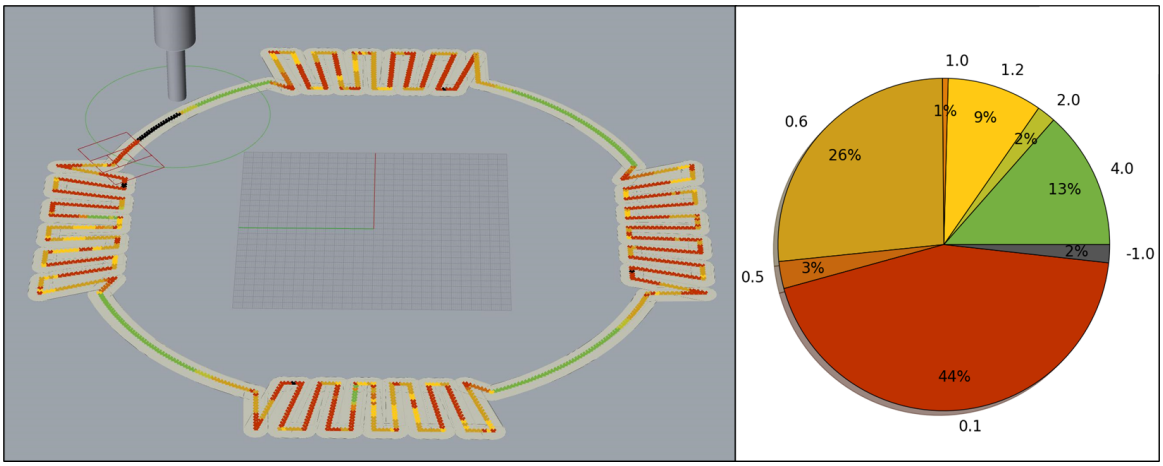

FIGURE 4.4: Quality of the Weaving Pattern toolpath with Point Evaluation

In 4.4 the Weaving pattern is represented. As can be seen, the percentage of points never observed (value equals -1) is very low (2%). This kind of toolpath has some difficult features to manage for the algorithm. It has no jumps but frequent direction changes, and few basic shapes such as continuous curved lines or even straight lines. But rather it has many short straight lines in directions orthogonal to each other. So the algorithm struggles a lot to compute a smooth path and to keep the points in an optimal position in the FOV. Although this is an important shape to be able to manage, the result is therefore delusional. the majority of points are inspected in a non-optimal way, with a wide amount being observed only in a peripheral region of the FOV. Therefore this product is rated as **Autonomus** but with **Low Quality** of inspection possible, with 15% of points framed in the ideal way).

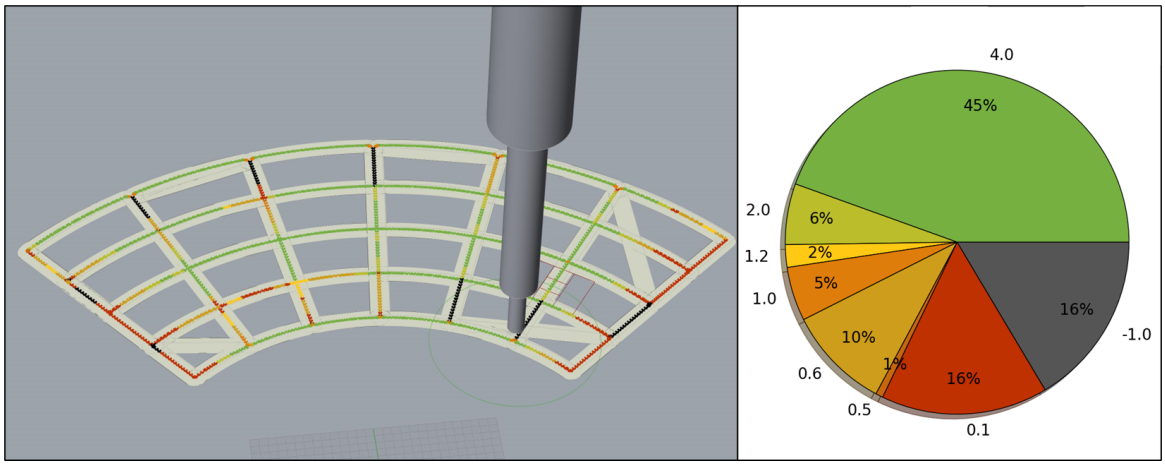

FIGURE 4.5: Quality of the Bent Grid toolpath with Point Evaluation

In 4.5 the Bent Grid pattern is represented. The process is **Not Autonoumus**, even for a small amount, in fact, the total amount of viewed points is 79%. Besides this, the amount of the points with good quality over the points observed is 64% so a **Sufficient Quality** can be guaranteed. This toolpath is similar to a product shown in 1.1 which is an example of what the machine is designed to manufacture.

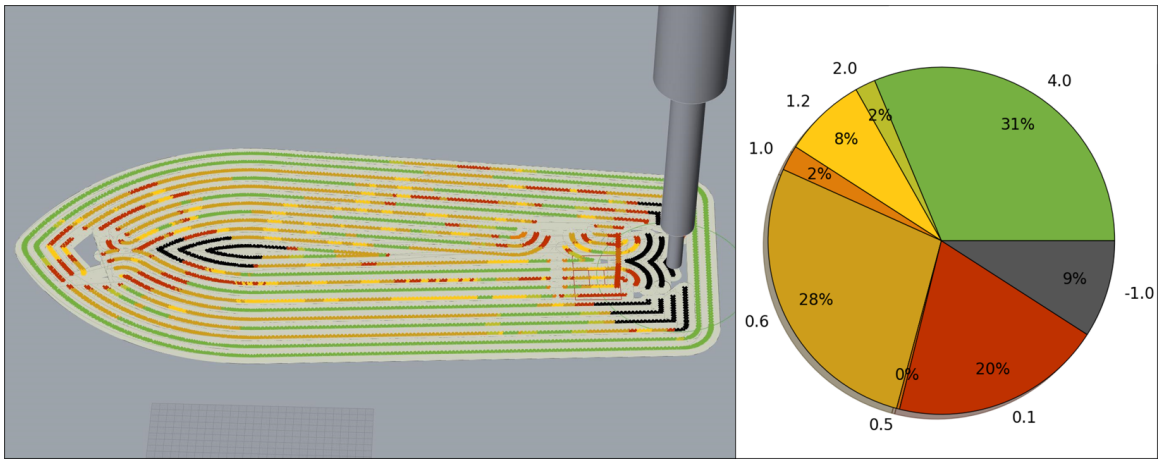

FIGURE 4.6: Quality of the 1-st layer of benchy toolpath with Point Evaluation

In 4.6 pattern of Benchy is represented. As previously said, it's unlikely that the M3DP machine will be used to manufacture this kind of toolpath. But still, this is a toolpath that has some features such as jumps, and direction changes, Thus some useful information about the performances of the algorithm can be derived from this. The process is **Semi-Autonomous**, it has just the 9% of points not observable. In this specific situation, this could be bad because the points are internal to the outer border of the figure, which means that the metal will spread towards the insides and could build up some unwanted metal in the center during the print. And not being able to inspect and prevent this kind of situation is not ideal. Besides this, the amount of the points with a good quality over the points observed is 36% so a **Low**

Quality can be guaranteed. Another thing that can be pointed out is a large number of band-aligned-discont points.

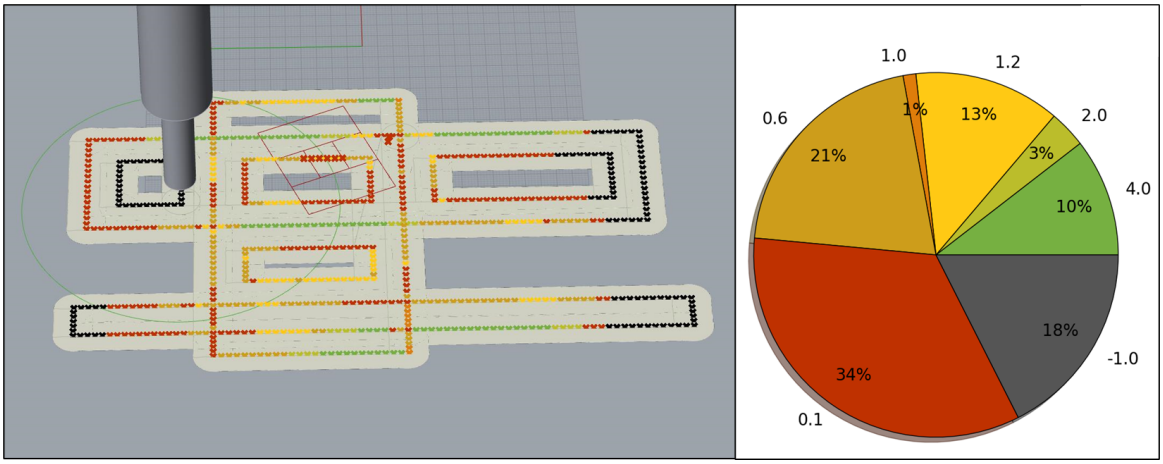

FIGURE 4.7: Quality of the rectangles toolpath with Point Evaluation

In 4.7 the Rectangles pattern is represented. The process is **Semi-Autonomous** as the total amount of viewed points is 82%. Although the quality of the inspection is quite poor: 15% this kind of shapes with narrow and squared curves are considered **Low Quality**.

4.2.2 Results with point evaluation and safe area

With the safe area option enabled, it was expected to produce promising results and decrease back-checking occurrences. The time saved without inspecting the same points already observed, the camera should have positioned itself in more convenient locations to better exploit its domain by moving across it more smartly.

In figure 4.8 the toolpaths are compared when inspected with and without the safe area feature. For each toolpath can be observed that the performance is almost the same.

4.3 Simulation environment

The results listed so far are certainly relevant to the project, but another result has been the production of the simulation environment and the algorithm in Python and Grasshopper.

Through the dynamic environment of Grasshopper, it is possible to load a .nc file into the system and obtain, after a reasonable calculation time, a simulation of the first layer of each product that can be consulted and explored point-wise. The script also shows the quality of the inspection of each point intuitively and understandably as shown in images from 4.2 to 4.8. Highlighting which points are not observable, and which ones are better. The simulation environment could also be used in the future for a deeper study of these topics and possibly improved, for example by integrating a functionality for the upper layers. Or by refining the calculation algorithm of the camera path, also being able to make comparisons between the previous algorithm and the one after optimization.

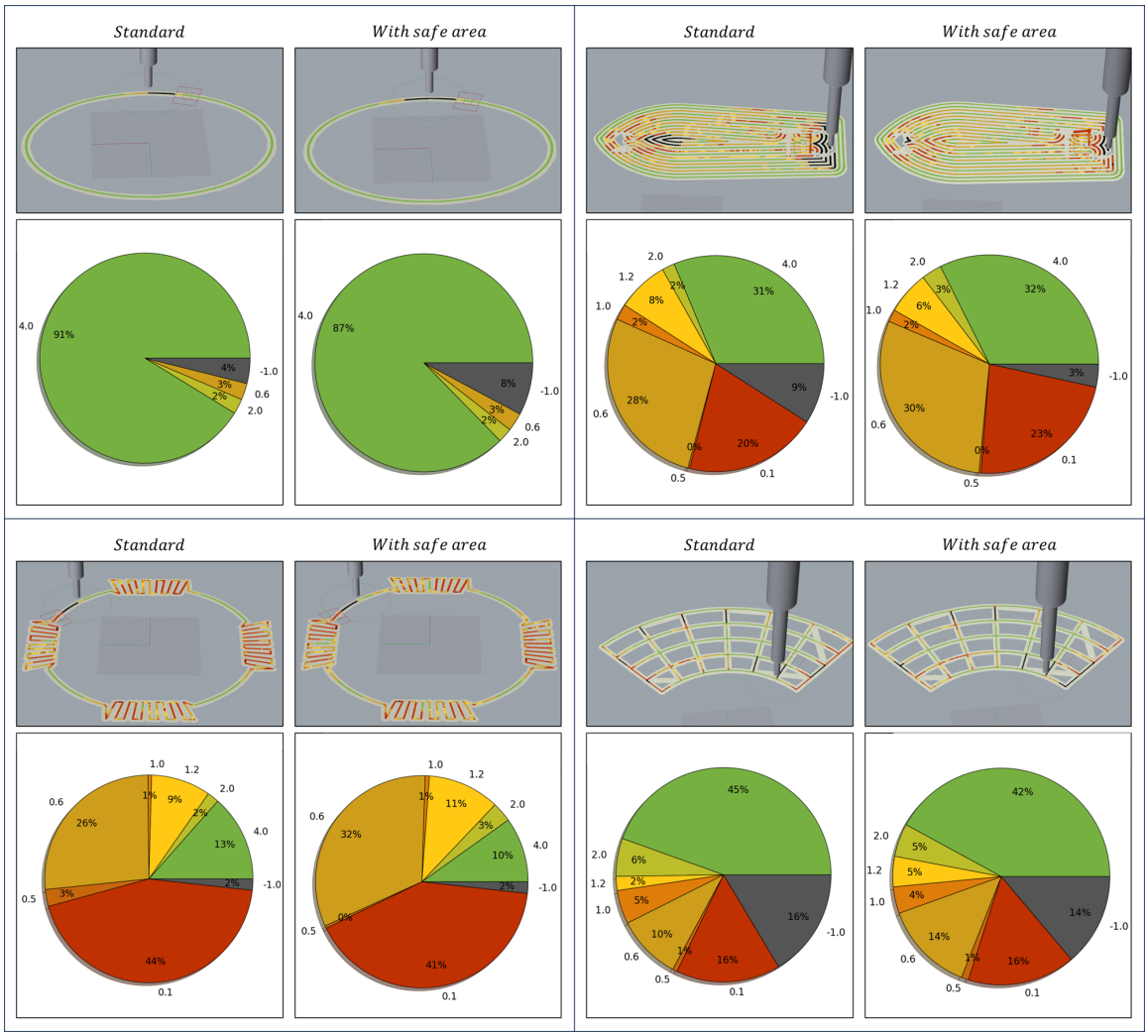

FIGURE 4.8: In this figure are shown some of the previously discussed toolpaths and their performance with and without the safe area feature.

Chapter 5

Conclusions

5.1 Simulation performances

After evaluating and commenting on the performance of the path planning algorithm in the previous section, and given the evaluation criteria explained at the beginning of 4, some conclusions can be drawn about the feasibility of this portion of the project: the path planning of the vision system aimed at three-dimensional analysis of the deposited titanium filament. The results show that with the constraints imposed by the premises, the results of an inspection would be positive and promising for very simple geometric figures such as cylinders, tubes or products that can be broken down into many wide curved lines with a high radius value. For products composed of angular shapes, with frequent changes of direction during manufacturing, the situation changes. It becomes very difficult for the vision system to follow the path.

The M3DP machine has remarkable capabilities as it can move on 5 axes (possibly 6). To produce a visual inspection system that can handle the complexity of movements along all these axes and at the same time avoid creating collisions and maintain good inspection performance, it is necessary to investigate the problem more deeply and this thesis can therefore be useful to highlight some of the main criticalities that the problem exposes. In the constraints stated in 3.2.2 The problem had already been simplified from five to three axes; after that, the problem was further simplified by avoiding the use of the Z axis, at least temporarily, to better study the behavior of the machine one layer at a time. After studying this simplified situation, it is clear that the best results continue to be those in which only one axis is moved (A-axis) or those in which two axes (X and Y) are used to simulate a circular movement. Therefore, one of the conclusions that can be drawn is that to have a reliable inspection with the constraints that have been imposed, a large part of the movement flexibility that this machine possesses has to be given up.
The situation could change if any of the constraints would loosen up. For example, as explained later in 6.4 the best alternative to optimize the inspection quality without increasing the energy consumption would be to have a vision system which works independently of the torch. Thus a second independent robot would be the best solution despite the algorithm complexity increase.

5.2 Safe Area improvement performance

The Safe Area feature that was supposed to improve the performance of the algorithm, by just visually judging from the animations, manages to partially prevent the back-checking circumstance. However, it does not result in a significant performance improvement. Indeed, judging from the current situation of the algorithm,

the result is slightly worse when the feature is active compared to when it is not. Therefore, in the future, some time can be spent improving the behavior of this feature in order to bring about an actual improvement in its effectiveness.

Chapter 6

Future developments

Due to the amount of time allowed for the writing of this thesis, some aspects identified during the initial problem study phase have been simplified or temporarily ignored as they were momentarily secondary, given a future deepening. Below are some of these aspects, along with some ideas for improvements and future developments.

6.1 Upcoming stage improvements

Path planning algorithm design improvement
The algorithm can be improved by reorganizing the structure, taking more time to learn about the simulation software Rhinoceros and Grasshopper, and researching and brainstorming about necessities and the actual project-imposed constraints of the problem. Right now, the algorithm works with one-layer prints, but even with one-layer prints, if the layer is complex (e.g. has more than 7000 coordinates), the performance in terms of speed lowers, and it's therefore not scalable.

Back Checking simple detection
A rather easy-to-implement idea is to try to identify which indices of the camera coordinates are those in which back checking occurs and treat those areas as dark zones in order to take advantage of the back checking instants to go towards the first useful position after it like it is done with the safe area solution.

Visual notification
A possible feature to add could be to notify through a signal in the footage that the frames displayed do not represent novel points but rather points already seen, for example through a red light or simply with a software flag on this kind of frame.

Toolpath Back Checking factor
One useful feature to add is some sort of control on how often the back-checking occurs. Computing this back-checking factor would give an overview of how much the back-checking is an issue in a specific toolpath, and if it's worth performing additional calculations to avoid it or if instead is not

Make good use of promising scores
It is frequent to have a point score of 0.6, ocra-yellow in charts, as shown in table 6.1. This score is given when a point is observed in the central band of the FOV but not in the central area, it is aligned towards the right orientation but is not considered to continue to the previous and subsequent ones. This can be caused by pieces of the seam that enter the FOV randomly and are not therefore important to be observed.

TABLE 6.1: Percentages of 1.2 (center-aligned-discont) and 1.2 (band-aligned-discont) scores in the analyzed toolpaths

Figure	% of 1.2 score	% of 0.6 score	Total improvement
4.4	9	26	35%
4.5	2	10	12%
4.6	8	28	36%
4.7	13	21	34%

But some other times, they are valid points and are just in tricky positions so a continuity cannot be identified but they could provide useful information once observed with the ICI algorithm. Another useful score that can be worked on to be considered a valid one, is 1.2, bright yellow in charts, which occurs less often with respect to the other score. These points are not only aligned but also centered in the FOV, the issue is still the continuity, which would be the parameter to improve and master during the development of this feature.

Toolpath splitting

As proposed in 3.3.5 an optimal solution would be the one of splitting the toolpath into primitive shapes such as straight lines and curves. Once the slices of path are ready, for each kind of primitive the algorithm would know how to behave at best to observe the slice. For example: for straight and curved lines it can modify the .nc file in order to add "dummy" coordinates where the torch will not deposit metal but would make the unreachable points of the toolpath observable by the camera. An approach towards this solution is proposed in [19]. With the implementation of this technique also the missing ends problem would be solvable.

Among the primitives, a weaving pattern like the one represented in figure 6.1 could then be added in order to treat that portion of the toolpath as a large and slow curved line, to be able to record the full width of the pattern in the camera view and treat it as a single thick seam of metal.

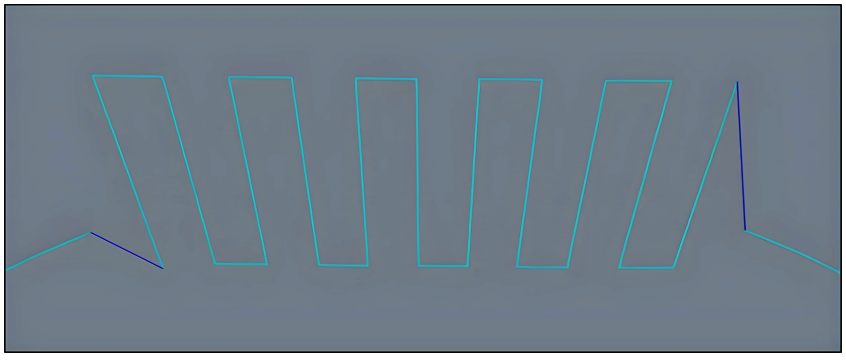

FIGURE 6.1: Example of a weaving pattern template. By extracting the key features of a template like this, in a future development, it could be possible to identify the difficult portions of the toolpath and simplify them to make the camera follow a more gentle and predictable movement.

This solution should also consider the geometry and volume in which the torch can move, without colliding the arm that holds the camera with the outer frame of

the machine, with other cameras, and with the product that is being manufactured. Moreover, it would greatly increase energy consumption as the torch would have to be turned off and brought back to temperature a number of times far greater than predicted by the original toolpath.

Virtual footage analysis

As explained in 3.3.5, it is possible to produce a "virtual footage" of the inspection process from the Point of View of the camera. This crafted footage in the future could be analyzed with some sort of ICI algorithm adaptation to simulate how well the 3d reconstruction could work on a specific toolpath. In this way, there could also be a virtual sample to compare the inline 3D reconstruction with and perform a more accurate evaluation of how well the manufacturing process is going

Views of point counter for ICI

The ICI algorithm works by interpolating multiple views of the same points. Therefore, it would be useful to count how many times a specific point is seen. With this data, it could be possible to evaluate which portions of the toolpath are effectively inspectable by ICI.

6.2 middle stage improvements

Bracket design
It's necessary if it's decided to try a real implementation, to design a metal bracket that attaches to the torch and can rotate around it without coming across other cameras and cables mounted around the torch. The shape of the bracket will need to be optimized to have the lowest impact on the torch performance. For design purposes, The Machining Plugin from ModuleWorks offers a machine builder tool that can be used to simulate the machine as a 3d object and to move all the axes manually. Moreover, it allows the addition of shapes and models to the simulations and to pilot them through a script to simulate their attachment to the machine. this could be useful to see the volumetric impact and the behavior of the bracket around the torch.

Actuator choiche
Besides the design of the bracket, an electronic system must be designed, tested, and built, and proper actuators must be chosen in order to move in the right way and at the right time the camera around. Therefore a study about the topic must be pursued to find a working setup.

6.3 Late stage improvements

Multi layer prints
One of the first aspects on which a blind eye was turned on is the multi-layer products. Actually, almost all the products that the machine can be used to manufacture are multi-layer. Additive manufacturing most of the time relies on this key feature. Even though, to better understand the problem it has been necessary to simplify it

and one of the simplifications adopted was the one of focusing only on one layer at a time.

The goal was to develop a reliable and robust algorithm for a general layer and then apply it to all the layers in sequence. Of course, this is not a behavior that is available out-of-the-box, and therefore a future development would be to add a script that allows multiple layers to be analyzed and good multi-layer footage to be produced.

One big issue that stopped this part of the development was the awareness that a crafted multi-layer file.nc would probably look different from a proper file.nc used to perform an actual manufacture. This could be due to multiple small parameter adjustments, which, when added one to another, can result in a very different approach. Therefore, the robustness of the algorithm that would have to be developed would have been too high. Therefore, this portion of the problem has been postponed until it becomes a future development.

Path planning algorithm improvement: optimization
Due to the Rhino-Grasshopper 3D environment, the execution of the algorithm for large prints takes a long time even though only one-layer prints are allowed. This means that a useful development would be a focus on the scalability of the algorithm. Moreover right now in Grasshopper, the scripts are coded in Python, which is known to be not the ideal solution for the performance, it is in fact great for prototyping. In an optimization phase, some attention can also be put on the porting of the code from Python to c#. Grasshopper offers indeed a c# block for scripting. this already would reduce the computing time. Furthermore, another solution that can be adopted to decrease the computational time even more is the full porting of the software in c#. This solution would aim to provide only the coordinates for the camera as an output, saving all the time used to render the 3D environment.

Mixed online and offline approach
In the late stage of the project, when a first real implementation has already been realized, a mixed online and offline approach can be developed as suggested before in 3.2.5. This addition to the software can be considered as visual servoing as the trajectory of the camera would be adjusted in real-time based on a feedback loop provided by a sensor: the camera itself or any other inspection sensors. This addition would need to add to the machine software a task that monitors and performs the calculations of these sensors. Or rather, a microcontroller can be arranged in the proximity of the machine to perform all the calculations. A robust software should be designed and implemented along with it.

6.4 Alternative development ideas

Without movement constraints
The solutions considered so far are all focused on a one-robot approach. This means that the WAAM machine will be attached to another robotic addition which will be able to move as it is widely explained above. With this setting, however, the camera is constrained to the torch and its range of motion is dependent in each instant to the position of the torch movements. An alternative approach would be to make use of another robot, completely independent from the torch but aware of its presence in order to avoid collisions. For example, a robotic arm mounting a camera in its effector could do the job. In this case, however, the movements of the robot arm should

be highly precise in his movements, as the camera has to be moved guaranteeing a linear movement and avoiding wobbling and loss of focus.

Without inline constraint

The approaches described in this thesis are all oriented to a real-time inspection and evaluation of what is being inspected. In literature this is referred to as "inline quality inspection" [30, 31]. An alternative approach could be the offline inspection. In this case, there would not be some of the strictest constraints that we worked on for the entire thesis. If the inline constraint is removed, in fact, once one layer is finished, the torch would walk again across the path of the layer, but this time focusing only on giving the camera the best path possible to record clean footage. Therefore a new algorithm to compute the camera path must be programmed. This time although it would be much easier. The second camera axis would play a crucial role in keeping the footage stable and oriented in the right direction. This approach could be mixed with toolpath splitting at 6.1. In this way, good and optimal coverage would be possible on many kinds of primitives. And also the problem of recording the beginning and the end of each split apart of the toolpath would be solvable.

The downside of this approach is that manufacturing processes of this kind aim at keeping the work time as low as possible, either for the high requests of customers to use these machines or to waste the least amount of energy. During the manufacturing process, in fact, the goal would be to print continuously, one layer after another. This way, the torch is more easily kept at the right temperature, and the titanium piece doesn't suffer from dilation and restriction stress caused by temperature variation. But if for each layer it would be necessary to stop and use quite a lot of time to inspect the layer, then the energy consumption rises along with the risks of faulty products and prints.

Bibliography

[1] Masoud Shaloo et al. "A Review of Non-Destructive Testing (NDT) Techniques for Defect Detection: Application to Fusion Welding and Future Wire Arc Additive Manufacturing Processes". In: *Materials* 15.10 (2022), p. 3697.

[2] V Schmitz, M Kröning, and KJ Kangenberg. "Quantitative NDT by 3D image reconstruction". In: (1996), pp. 735–744.

[3] Annamaria Gisario et al. "Metal additive manufacturing in the commercial aviation industry: A review". In: *Journal of Manufacturing Systems* 53 (2019), pp. 124–149.

[4] Malin Lervåg et al. "Additive manufacturing with superduplex stainless steel wire by cmt process". In: *Metals* 10.2 (2020), p. 272.

[5] Jyrki Savolainen and Mikael Collan. "How additive manufacturing technology changes business models?–review of literature". In: *Additive manufacturing* 32 (2020), p. 101070.

[6] Joel C Najmon, Sajjad Raeisi, and Andres Tovar. "Review of additive manufacturing technologies and applications in the aerospace industry". In: *Additive manufacturing for the aerospace industry* (2019), pp. 7–31.

[7] Adrita Dass and Atieh Moridi. "State of the art in directed energy deposition: From additive manufacturing to materials design". In: *Coatings* 9.7 (2019), p. 418.

[8] Terry Wohlers et al. "Wohlers report 2019: 3D printing and additive manufacturing state of the industry". In: *(No Title)* (2019).

[9] Bintao Wu et al. "A review of the wire arc additive manufacturing of metals: properties, defects and quality improvement". In: *Journal of manufacturing processes* 35 (2018), pp. 127–139.

[10] Tuhin Mukherjee, W Zhang, and Tarasankar DebRoy. "An improved prediction of residual stresses and distortion in additive manufacturing". In: *Computational Materials Science* 126 (2017), pp. 360–372.

[11] Davoud Jafari, Tom HJ Vaneker, and Ian Gibson. "Wire and arc additive manufacturing: Opportunities and challenges to control the quality and accuracy of manufactured parts". In: *Materials & Design* 202 (2021), p. 109471.

[12] Zai-Gen Wu et al. "Inline inspection with an industrial robot (IIIR) for mass-customization production line". In: *Sensors* 20.11 (2020), p. 3008.

[13] Qianru Wu et al. "Effect of molten pool size on microstructure and tensile properties of wire arc additive manufacturing of Ti-6Al-4V alloy". In: *Materials* 10.7 (2017), p. 749.

[14] Vanessa Staderini et al. "Surface sampling for optimal viewpoint generation". In: *2023 IEEE 13th International Conference on Pattern Recognition Systems (ICPRS)*. IEEE. 2023, pp. 1–7.

[15] Svorad Stolc, Petra Thanner, and Markus Clabian. "Inline Computational Imaging: single sensor technology for simultaneous 2D and 3D high definition inline inspection". In: *Imaging and Machine Vision Europe* 89 (2018), pp. 34–35.

[16] Bernhard Blaschitz et al. "High-speed inline computational imaging for area scan cameras". In: (2021).

[17] JG Mercado Rojas et al. "Plasma transferred arc additive manufacturing of Nickel metal matrix composites". In: *Manufacturing letters* 18 (2018), pp. 31–34.

[18] Ruben Bayu Kristiawan et al. "A review on the fused deposition modeling (FDM) 3D printing: Filament processing, materials, and printing parameters". In: *Open Engineering* 11.1 (2021), pp. 639–649. DOI: doi:10.1515/eng-2021-0063. URL: https://doi.org/10.1515/eng-2021-0063.

[19] Giuseppe Venturini et al. "Feature based three axes computer aided manufacturing software for wire arc additive manufacturing dedicated to thin walled components". In: *Additive Manufacturing* 22 (2018), pp. 643–657. ISSN: 2214-8604. DOI: https://doi.org/10.1016/j.addma.2018.06.013. URL: https://www.sciencedirect.com/science/article/pii/S2214860418301684.

[20] Eva M Perez-Soriano et al. "Processing by additive manufacturing based on plasma transferred arc of hastelloy in air and argon atmosphere". In: *Metals* 10.2 (2020), p. 200.

[21] Timothy S Newman and Anil K Jain. "A survey of automated visual inspection". In: *Computer vision and image understanding* 61.2 (1995), pp. 231–262.

[22] A Noble et al. "Template guided visual inspection". In: (1992), pp. 893–901.

[23] Christopher C Yang, Michael M Marefat, and Rangasami L Kashyap. "Active visual inspection based on CAD models". In: *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE. 1994, pp. 1120–1125.

[24] Lukas Traxler et al. "Experimental comparison of optical inline 3D measurement and inspection systems". In: *IEEE Access* 9 (2021), pp. 53952–53963.

[25] Laurin Ginner, Simon Breuss, and Lukas Traxler. "Fast Inline Microscopic Computational Imaging". In: *Sensors* 22.18 (2022), p. 7038.

[26] Vo Duy Cong and Le Duc Hanh. "A review and performance comparison of visual servoing controls". In: *International Journal of Intelligent Robotics and Applications* 7.1 (2023), pp. 65–90.

[27] 2022. URL: https://www.food4rhino.com/en/app/animation.

[28] Robert McNeel Associates. *Rhinoscriptsyntax in python*. URL: https://developer.rhino3d.com/guides/rhinopython/python-rhinoscriptsyntax-introduction/.

[29] URL: https://github.com/stgeorges/rhinopython/blob/master/scripts/rhinoscript/geometry.py.

[30] Victor Azamfirei, Foivos Psarommatis, and Yvonne Lagrosen. "Application of automation for in-line quality inspection, a zero-defect manufacturing approach". In: *Journal of Manufacturing Systems* 67 (2023), pp. 1–22. ISSN: 0278-6125. DOI: https://doi.org/10.1016/j.jmsy.2022.12.010. URL: https://www.sciencedirect.com/science/article/pii/S0278612522002291.

[31] Alexander Pierer et al. "Zero-error-production through inline-quality control of presshardened automotive parts by multi-camera systems". In: 1157.1 (2021), p. 012074.