8-2019

Toolpath Planning Methodology for Multi-Gantry Fused Filament Fabrication 3D Printing

Hieu Trung Bui
University of Arkansas, Fayetteville

Toolpath Planning Methodology for Multi-Gantry Fused Filament Fabrication 3D Printing

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Industrial Engineering

by

Hieu Bui
University of Arkansas
Bachelor of Science in Industrial Engineering, 2015
University of Arkansas
Master of Science in Operations Management, 2017

August 2019
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

_____
Harry Pierson, Ph.D.
Thesis Director

_____
Sarah Nurre, Ph.D.
Committee member

_____
Kelly Sullivan, Ph.D.
Committee member

Abstract

Additive manufacturing (AM) has revolutionized the way industries manufacture and prototype products. Fused filament fabrication (FFF) is one of the most popular processes in AM as it is inexpensive, requires low maintenance, and has high material utilization. However, the biggest drawback that prevents FFF printing from being widely implemented in large-scale production is the cycle time. The most practical approach is to allow multiple collaborating printheads to work simultaneously on different parts of the same object. However, little research has been introduced to support the aforementioned approach. Hence a new toolpath planning methodology is proposed in this paper. The objectives are to create a collision-free toolpath for each printhead while maintaining the mechanical performance of the printed model. The proposed method utilizes the Tabu Search heuristic and a combination of two subroutines: collision checking and collision resolution (TS-CCR). A computer simulation was used to compare the performance of the proposed method with the industry-standard approach in terms of cycle time. Physical experimentation is conducted to validate the mechanical strength of the TS-CCR specimens. The experiment also validated that the proposed toolpath can be executed on a custom multi-gantry setup without a collision. Experimental results indicated that the proposed TS-CCR can create toolpaths with shorter makespans than the current standard approach while achieving better ultimate tensile strength (UTS). This research represents opportunities for developing general toolpath planning for concurrent 3D printing.

Acknowledgements

I am grateful to my advisor, Dr. Harry Pierson, for his continuous support and guidance. His expertise has helped me explore new objectives for this research. I would like to thank Dr. Sarah Nurre, and Dr. Kelly Sullivan for their valuable time and support throughout the research. I also want to thank Mr. Mark Kuss for his help in using the machine to measure the tensile strength.

Table of Contents

1. Introduction 1
2. Literature review 3
   2.1. Toolpath planning optimization for FFF 3
   2.2. Concurrent 3D printing 4
   2.3. Multi-object motion planning 4
   2.4. Summary 5
3. Methods 6
   3.1. Overview 6
   3.2. Tabu search heuristic 7
   3.3. CCR subroutines 11
      3.3.1. Collision checking subroutine 14
      3.3.2. Collision resolution subroutine 15
   3.4. Closed-loop control and resynchronization process. 17
   3.5. Experimentation 20
      3.5.1. Simulation setup 20
      3.5.2. Custom multi-gantry printer's setup 22
      3.5.3. Tensile specimen design 23
      3.5.4. Tensile testing procedures 25
4. Results 26
   4.1. Simulation results 26
   4.2. Physical experiment results 27
      4.2.1. Results from tensile test 27
      4.2.2. TS-CCR and closed-loop control validation 29
5. Discussions 31
   5.1. Performance of TS-CCR 31
   5.2. Physical test 34
6. Conclusions 35
7. References: 36
8. Appendix 38
   8.1. Collision resolution subroutine illustration. 38

List of Tables

Table 3.1. Process Parameters for 3D printed parts 24
Table 4.1. Results from tensile testing. 28

List of Figures

Figure 1.1. Toolpath illustrations of applying OSA and the proposed method on "IE Hog" layer on 2-gantry 3D printer. 2

Figure 3.1 Raster segments with identification number of IE Hog layer (2% infill was used for demonstration purpose) 7

Figure 3.2. Illustration of the three operators used to generate new solutions 8

Figure 3.3. Flowchart of the tabu search algorithm 9

Figure 3.4. Flowchart of the solution evaluation with the help of CCR 11

Figure 3.5. Toolpath representation of a potential solution. 12

Figure 3.6. Illustration of the calculation of the x-coordinate of each gantry. 14

Figure 3.7. Trajectory plot of the toolpath in Figure 3.5, collision checking subroutine identified 6 potential collisions 15

Figure 3.8. Trajectory plot result from the CCR. Several delays were added to solve the 6 collisions in Figure 3.6 17

Figure 3.9. Flowchart of closed-loop control with the resynchronization process built-in 19

Figure 3.10. a) layer from IE Hog; b) layer from airfoil frame [21]; c) layer from topology optimized bracket [22]; d) layer from rotor hub wind turbine nose [23] 21

Figure 3.11. a) standalone Makergear M2 3D printer; b) the custom setup comprised of two Makergear printers to test the performance of the closed-loop control. 22

Figure 3.12. a, b) printed part of the specimens using two different approaches; c) close up shot of the seam; and d) complete specimen. 24

Figure 3.13. a) MTS tensile testing machine; b) mechanical wedge grips 25

Figure 4.1. Makespan comparison of four selected layers at 30% infill, where the proposed TS-CCR can yield solutions with shorter makespan than solution obtained from OSA. 26

Figure 4.2. Stress-strain plot for specimens printed with a) OSA method, b) TS-CCR method. 28

Figure 4.3. Fractures of specimens printed by two different methods. 28

Figure 4.4. Finished layers 29

Figure 4.5. The custom setup consists of two printers, each printing the assigned part. While printing, the two heated bed never touching each other. The safety distance (yellow shaded region) were violated twice result in 2 resync actions 30

Figure 4.6. a) trajectory plot generated from the TS-CCR for "IE hog" in **Figure 4.4** (zoomed section shows that the two trajectories do not overlap); b) trajectory plot generated from the encoders, two resync actions allow the gantries synchronized throughout the print. These two resync actions result in 3 seconds different between two trajectory plots (note that the homing duration was excluded when calculating the actual makespan) 30

1. Introduction

Additive manufacturing (AM), also known as 3D printing, is a process of creating three-dimensional objects one layer at a time based on a computer-aided design (CAD) model. AM can produce complex geometric objects that are unachievable or require a combination of several traditional manufacturing processes. Fused filament fabrication (FFF) is an AM process that creates a model by extruding thermoplastic material. FFF is widely available and used in many applications because of its affordable cost and ease of use. The applications include prototypes, customized functional models, tooling and moldings, and end-use parts. Since it was first introduced in the 1980s, there have been many significant improvements on FFF technology. These improvements have many aspects from faster printing speed to the use of a wide array of printable materials [1-3]. Despite these improvements, the fundamental physical constraints still limit the FFF's speed. The FFF process is limited by how fast the printhead can move, melt, and dispense the thermoplastic material. Several approaches have been introduced to mitigate this limitation, one of which is multi-gantry FFF 3D printing. This printing concept, Project Escher, was introduced by Autodesk which allows multiple collaborating toolheads to work simultaneously on a single build. This concept relies on the toolpath planning that was implemented in a Netfabb software to distribute toolpath between multiple printheads. While having more printheads can theoretically reduce the build time (makespan), this approach has its own drawbacks. The opportunity for collision increases as more printheads are added to the configuration. To avoid collision, the toolpath planning algorithm in Netfabb software divides each layer into sub-regions orthogonal to the axis of gantry travel and assigns them to the gantries. This toolpath planning approach will be called orthogonal segmentation approach (OSA) in this research. Each gantry prints the associated sub-regions in a left-to-right sequence.

A seam represents the joining of two sub-regions where the outer-most perimeters are touching each other. The result of a layer sliced with the OSA can be seen in Figure 1.1. The Netfabb's solution ensures all the gantries are perfectly asynchronized to avoid collisions. A preliminary study of the OSA performance was conducted to identify the potential for improvement. Various seams were created across the layer and they would create weak areas that adversely affect the strength of the printed model. In addition, the results also showed that the OSA's makespan is larger than the theoretical minimum that the machine could achieve. Although the OSA is computationally efficient and easy to implement, it leaves room for further improvement.

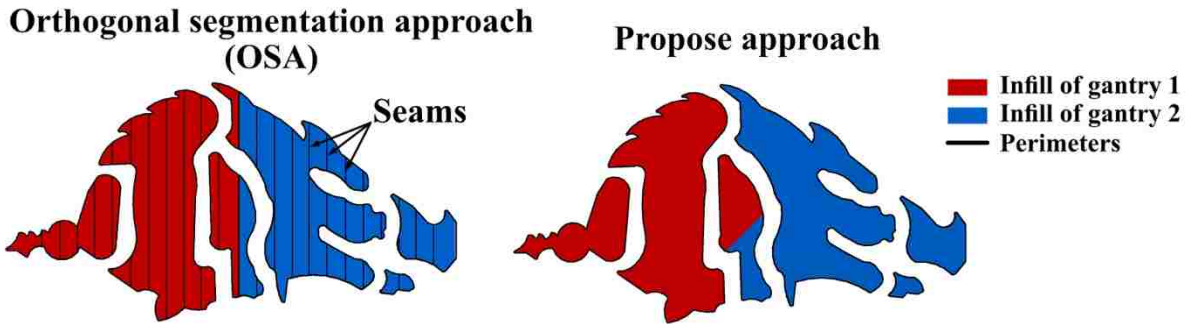


Figure 1.1. Toolpath illustrations of applying OSA and the proposed method on "IE Hog" layer on 2-gantry 3D printer.

The potential for improvement and the lack of alternative path planning approach for multi-gantry FFF 3D printer inspired the development of a new methodology. The objective of the proposed method is to minimize a single layer makespan while considering collision constraints for 2-gantry printers. Although the multi-gantry system is one of the several kinematic configurations of concurrent FFF 3D printing, the finding of this work can provide insights into the development of generalized multi-tool path planning problem for AM process.

2. Literature review

A literature survey was conducted to study the work done in the subject of (1) toolpath optimization approaches for FFF 3D printing, (2) concurrent 3D printing, and (3) multi-object motion planning.

2.1. Toolpath planning optimization for FFF

Most research on optimizing nozzle path planning for FFF has formulated the printhead path planning problem as traveling salesman problem [4, 5], undirected rural postman problems (URPP) [6], or some variant of TSP and applied heuristic and approximation algorithms to solve the problem. Volpato et al. [4] presented two optimization algorithms to reduce the repositioning distances: (1) nearest neighbor procedure and (2) a combination of the nearest and the farthest insertion method. The results indicated that the improvement of using optimization in AM path planning is considerable. The results also showed that depending on the geometry, different methods provide the best results. Fok et al. [6] formulated the path planning problem as a URPP and applied an extended ant colony optimization (ACO) algorithm to shorten the printing process by eliminating unnecessary movement of the printhead. The proposed method adaptively adjusts the number of iterations in ACO based on the similarity of the current and the previous optimized layers. Thus, the proposed method can accelerate the optimization and printing processes while still maintaining the quality of the solution. Wah et al. [7] developed a strategy that incorporates both Asymmetric Travel Salesman Problem (ATSP) and Integer Programming heuristic to solve the path planning problem. In addition, they implemented 2-opt and 3-opt local optimization within the Genetic Algorithm. The results from their testing of both strategies showed the reduction up to 50% in the repositioning distances, thus reducing the time spent from performing these jumps. Even though the time saving in each layer is small, such savings can

reduce the total fabrication time significantly (i.e., parts can require hundreds or thousands of layers to be fabricated). Wojcik et al. [8] proposed the use of modified zig-zag (MZZ) algorithm as the path generating algorithm and a genetic algorithm (GA) as the path optimization algorithm. The authors mentioned the importance of determining the parameters (i.e., number of epoch) for different layer size. In general, all of the mentioned toolpath planning optimization approaches are still limited to single printhead machines where they do not address collision constraints or decisions that affect mechanical integrity.

2.2. Concurrent 3D printing

Concurrent 3D printing is still a relatively new concept in AM. Yin et al. [9, 10] developed two heuristics to allocate toolpaths created by existing slicing software to available printheads with collision constraints. The simulation results for three printheads showed as much as a 60% reduction in printing time compared with single printhead printers. The method also demonstrated a technique to determine the optimal number of printheads for a specific layer. Zhang et al. [11] demonstrated the use of multiple mobile robots that work in a team to print a large concrete structure for the Building and Construction industry. Experimental results showed the proposed system is flexible and has high scalability and efficiency. Choi et al. [12] proposed an approach to concurrent toolpath planning for a multi-material layered manufacturing (MMLM) process. The approach applied a dynamic priority scheme to adjust the motion of the tools when a potential collision is detected.

2.3. Multi-object motion planning

The problem of coordinating multiple moving objects that work in a shared workspace without collision is covered extensively in various applications such as mobile robots, manipulation of robot arms, etc. [13-16]. In each application, various methods have been

introduced in which avoiding collision is a significant component. The similarity of these methods is that the potential collisions are detected at the beginning, then the motion of corresponding objects are subsequently coordinated to avoid collision. There are two main categories of the multi-object motion planning problem: (1) coupled method, and (2) decoupled method. In coupled methods, the configuration spaces of all object are combined into one unified configuration space, and the feasible path is then searched. On the other hand, in the decoupled method, the path of each object is individually created and subsequently organized to avoid collision [17]. Several decoupled methods have been introduced, such as adjusting the trajectory, inserting time delay, and modifying the velocity of objects [18]. The printheads or gantries do not have a collision-sensing capability, self-control, or a way to communicate with each other. They are controlled by a central controller where all actions are predefined before executing. Furthermore, the printhead in FFF should not vary in its speed as it can print inconsistent in road width, which allows air void to get trapped inside and compromises the structural integrity of the 3D printed part. In conclusion, applying heuristic optimization to generate collision-free toolpath by adjusting the toolpath and inserting time delay to each printhead fits in the multi-gantry FFF application.

2.4. Summary

The current literature on path planning optimization for FFF applies various heuristic techniques to minimize the total repositioning distance. However, it only focuses on a single toolhead configuration, hence no strategy is required for collision-avoidance. The literature on multi-object path planning is covered extensively but does not adequately provide information for solving the concurrent FFF path planning problem. The concurrent 3D printing process is still a new concept and requires more development. This gap motivates the development of:

- New toolpath planning methods for multi-gantry FFF printers where the objective is to minimize makespan subject to collision constraints while achieving good mechanical properties.
- The optimization procedure that uses a combination of heuristic approaches and collision avoidance subroutines to obtain near-optimal toolpath solutions for a 2-gantry printer.

3. Methods

3.1. Overview

The objective of the proposed method is to minimize the printing time, also referred to as makespan, of the infill portion of a layer. The makespan traditionally includes two components: (1) the time spent on dispensing the material, and (2) the time spent performing rapid travel to a new location. In multi-gantry FFF printing, the time spent responding to a potential collision can also affect the makespan. The proposed method can be broken down into two problems. For example, given a number of infill raster segments (Figure 3.1); the first problem is assigning these segments to each printhead and the second problem is the order in which the printhead is required to print the assigned segments to avoid mutual collision (sequencing problem). To address the two problems mentioned, a tabu search (TS) heuristic was used to find a near-optimal solution. To evaluate the solutions in TS, two subroutines were developed: collision checking and collision resolution (TS-CCR). The information about the raster segments was extracted from the G-code file that was generated for a single printhead machine. This provided flexibility in controlling several parameters such as infill pattern, infill percentage, etc. Each raster segment is tagged with a unique identification number as shown in Figure 3.1.

Several assumptions were made in the development of this research. First, a two-gantry printer configuration was selected to model. However, the proposed method can be extended to

more than two gantries. For the ease of reading, left and right gantries mean the position of the gantries on the machine. Second, under the multi-gantry configuration, the gantries travel and only collide in the x-direction. Last, since the objective of this research was to find a collision-free toolpath that traversed all the raster segments for each printhead. The perimeters for the entire model were assumed to be printed by one printhead. Furthermore, additional benefits of printing the perimeter shells using one printhead are: (1) better surface quality, (2) no seams along the shell (for both aesthetic and strength reasons).

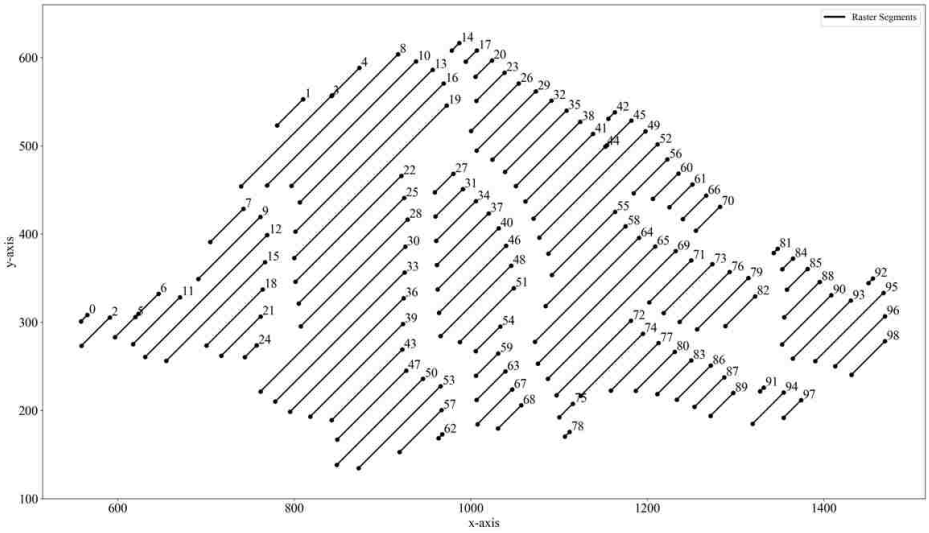


Figure 3.1 Raster segments with identification number of IE Hog layer (2% infill was used for demonstration purpose)

3.2. Tabu search heuristic

Tabu Search (TS) is a meta-heuristic that is capable of finding a good quality solution in a short running time [19]. TS uses memory to store information, use it to guide, and restrict the future search in order to obtain a better solution and to overcome local optimality.

The representation of a solution in TS is comprised of two lists. The first list includes the sequence that specifies the order in which raster segments are printed. Similarly, the second list includes the raster segments printed by the right printhead. To evaluate the quality of the

solution, the CCR subroutines, which will be discussed in the next section, is used. Specifically, the result from the CCR provides the infill makespan (smaller makespan means a better solution). At each iteration, new solutions are generated by applying one of the three operators. The first two operators are designed to enable improvements by modifying the (1) assignment and (2) sequencing decisions. First, the global swap (GS) operator exchanges two raster segments from the sequences of different printheads. This operator is crucial in deciding the best raster segment assignment for each printhead. Second, the local swap (LS) operator is used to swap two raster segment orders from the same sequence. By modifying the order in which the raster segments are printed, the LS might reduce the makespan by reducing the rapid movements required. Third, the rebalancing operator is used to allocate one raster segment from the sequence of the printhead that has a larger makespan to the end of the other sequence. Because adding delays to resolve collisions might adversely affect the makespan of one of the two printheads, the last operator is used to balance the makespan of the two printheads. Figure 3.2 and Figure 3.3 represent an illustration of the operators, and a flowchart of the proposed TS procedure, respectively.

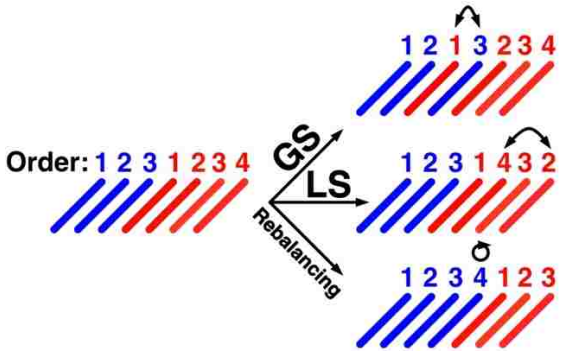


Figure 2.2. Illustration of the three operators used to generate new solutions

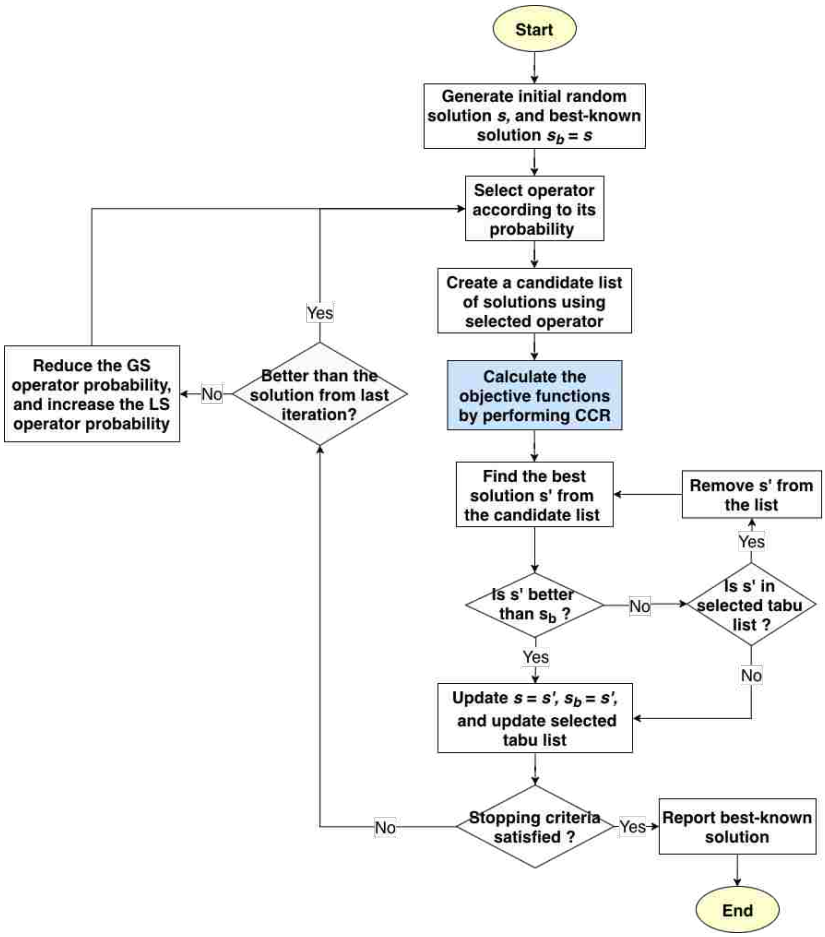


Figure 3.3. Flowchart of the tabu search algorithm

At the beginning of the algorithm, raster segments are randomly assigned to each printhead. The tabu search heuristic does not prohibit the solution with collisions but rather allows the CCR to resolve them. The infill makespan of the initial solution was expected to be high because of two factors: (1) there are a lot of chances for collision to happen, thus CCR would need to introduce long delays to resolve the problem, and (2) there is a high number of rapid movements since the sequences are not organized.

One operator is selected in each iteration based on its probability. At the early stage of the TS, the probability that the GS is selected is higher than the LS. After each iteration in which the GS cannot find an improved solution, the probability of it is slightly reduced (i.e., 0.5 %). As

the search continues, the LS operator slowly becomes more attractive. The probability of the TS choosing the rebalance operator is set relatively low. This means that the search occasionally checks for the opportunity to balance the infill makespan between the two toolpaths instead of doing so in every turn.

With the selected operator, each move applied to the current solution results in a new solution. The CCR is applied to evaluate the makespan for each solution in the candidate list, and the TS selects one that improves upon the current solution's makespan. If no move reduces makespan, the TS selects the solution with the least increase in makespan. This helps the search process reach to different regions. Every time a better solution is found, it will be updated as the best-known solution. The operator that created the selected solution is then labeled as a tabu. Tabus are used to prevent cycling, which means the solution that contains tabu's elements are banned. In other words, the tabu list prevents the algorithm from evaluating the same sequence over and over again. Tabu moves are stored in a short-term memory list, called tabu list. The moves in the tabu list can be removed using a first in first out method. One tabu list is designed for each of the three operators. A simple aspiration criterion was used. It allowed the tabu's move to be selected if it resulted in a solution with smaller infill makespan than the current best-known solution.

Two stopping criteria were used to terminate the TS. First, the TS terminates if the elapsed time at the end of an iteration exceeds a user-defined limit. Second, the TS also terminates, if the best-known solution is not improved above a certain threshold within another user-defined limit. For example, if a makespan of a layer is 20 minutes, and there is no improvement higher than 1% (12 seconds) for 5 minutes, the TS will terminate.

3.3. CCR subroutines

In the proposed method, the CCR is used to evaluate the makespan of the candidate solution. Figure 3.4 illustrates the solution evaluation process.

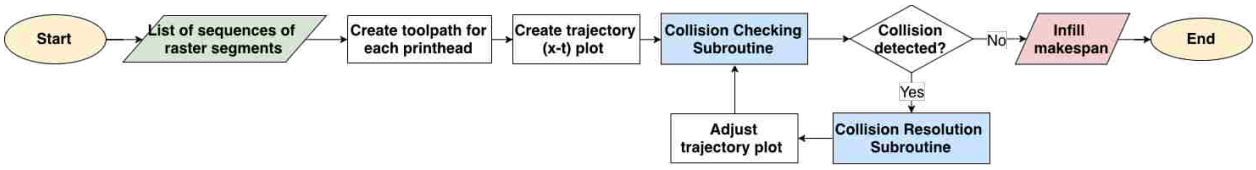


Figure 3.4. Flowchart of the solution evaluation with the help of CCR

The toolpath for each printhead is first constructed from the solution. The toolpath conversion procedure started from the beginning of the sequence, then depending on the location of the next raster segment, different types of connectors will be chosen accordingly. If the next segment is adjacent to the current segment, the two ends will be connected by a solid connector. This will mean the printhead will continue to dispense the filament when moving to the next segment. Otherwise, the two ends will be connected using a dashed connector, which means the printhead will stop extruding the filament and perform a fast maneuver to the next segment. The procedure continues until all the segments are visited. Figure 3.4 is an example that illustrates the toolpath of a potential solution. The representation of the solution in this example is:

$$s = \begin{bmatrix} \begin{bmatrix} 33, 36, 39, 43, 47, 50, 53, 57, 62, 46, 40, 37, 34, \\ 31, 27, 25, 22, 28, 21, 18, 15, 12, 9, 11, 5, 6, \\ 2, 0, 7, 3, 4, 8, 10, 13, 16, 19, 24, 30, 1 \end{bmatrix}, \\ \begin{bmatrix} 65, 69, 71, 73, 76, 79, 82, 86, 87, 89, 94, 97, 91, \\ 93, 90, 88, 85, 84, 81, 70, 66, 61, 60, 56, 55, 52, \\ 49, 44, 45, 42, 41, 38, 35, 32, 29, 26, 23, 20, 17, \\ 14, 48, 51, 58, 64, 54, 59, 63, 67, 68, 72, 74, 75, \\ 78, 77, 80, 83, 95, 92, 96, 98 \end{bmatrix} \end{bmatrix}$$

More specifically, the toolpath of the left gantry started printing from the raster segment #33 iteratively moving to segment #1. Similarly, the right gantry started printing segment #65 and ended at segment #98.

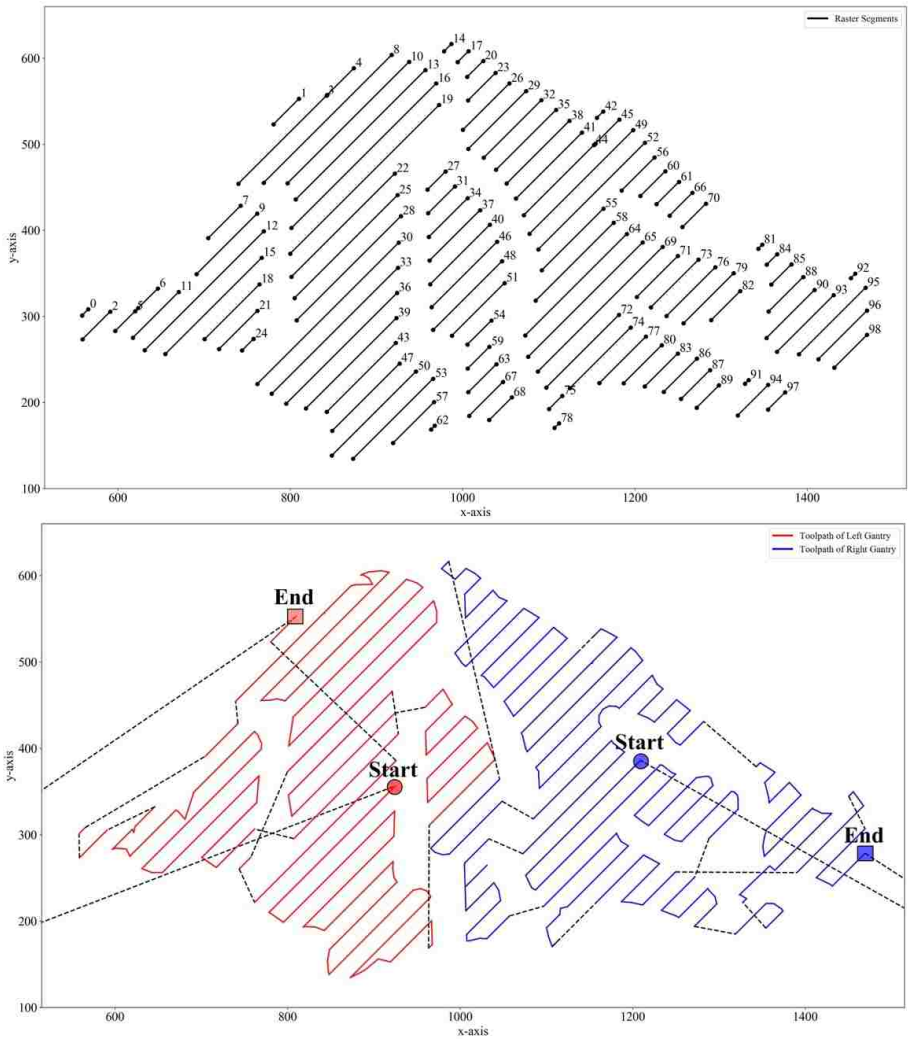


Figure 3.5. Toolpath representation of a potential solution.

It is important to separate the two different type of connectors since they have different operating speeds (printing speed vs. rapid movement speed). The time associated with completing each linear movement in the toolpath is then calculated using a trapezoidal velocity profile. Each movement has an entry speed, acceleration, operating speed, and deceleration values. At each movement, the printhead begins to accelerate from the entry speed to the

operating speed at a constant rate of acceleration. It then moves at the target speed for a certain distance and decelerates to the exit speed (i.e., entry speed of the next segment) at a constant rate of deceleration. The parameters such as acceleration, deceleration, and jerk are machine specific and need to be adjusted accordingly. This helps estimate the makespan of the toolpath as accurately as possible when executing this toolpath on the real machine. Note that the "jerk" used in most 3D printer firmware, Marlin, is not the same as "jerk" term in physics. The jerk, in this case, denotes the maximum instantaneous velocity change (i.e., measured in mm/s instead of mm/s^3). This value is used to determine the speed at the junction of two movements (a.k.a. the exit speed of the current movement and the entry speed of the next movement). Without jerk, the printhead needs to perform a complete stop in every corner, since it cannot begin accelerating in the new direction before arriving at the corner. Because the printhead cannot make instantaneous velocity change, the jerk will cause some vibrations to the machine. However, these vibrations are neglectable compared to the reduction in the printing time.

Since the gantries only collide in the x-direction, the collision-checking subroutine is much more straightforward than other configurations. In other words, a collision happens when two gantries share the same workspace at any moment in time. A trajectory plot of x-coordinates of the two gantries vs. time can be constructed to visualize collisions. A user-defined safety distance between two gantries is added to provide a buffer zone even though the collision is detected. The x-coordinate of each gantry is calculated by offsetting the x-coordinate of the printhead by θ, with the assumption that the printhead is located in the middle of the gantry. The equations with illustrations (Figure 3.5) of the x-coordinates of the left and right gantries are

$$x_{LG} = x_{LP} + \theta,$$

$$x_{RG} = x_{RP} - \theta,$$

$$\theta = \frac{(gantry\ width + safety\ distance)}{2},$$

Where x_{LG} and x_{RG} are the x-coordinates of the left and right *gantries*, respectively, and x_{LP} and x_{RP} are the x-coordinates of the left and right *printheads*, respectively.

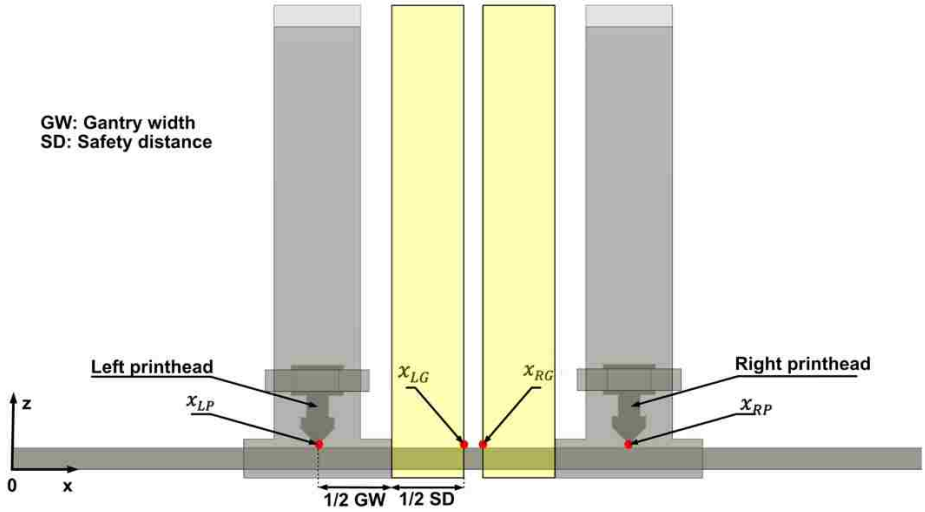


Figure 3.6. Illustration of the calculation of the x-coordinate of each gantry.

The collision checking subroutine either (1) concludes that the trajectories are collision-free, or (2) identifies the first collision in time. The collision resolution subroutine then resolves the first collision by introducing delay, but by doing so it might create more collisions at later times. Thus, by alternatively using both subroutines, they can remove all collisions.

3.3.1. Collision checking subroutine

The trajectory is comprised of paths that each gantry follows in the x-direction as a function of time. Feasible configurations must prohibit the gantries from occupying the same x-coordinate or passing through each other. With respect to the trajectory plot, this means that the left gantry trajectory must not fall below (i.e., its x-coordinate become smaller than) the right gantry trajectory. The collision checking subroutine starts from selecting the earliest segments at

time t = 0, one from each gantry. Antonio's algorithm [20] is applied to check if the selected pair of segments intersect. If there is an intersection, the associated segments are recorded as well as the intersection point. Otherwise, the trajectory segment with the earlier finishing time will be replaced by its next segment. The checking process continues until all pairs of segments are checked or the first collision is detected. In Figure 3.6, the plot indicates 6 potential collisions (shaded regions).

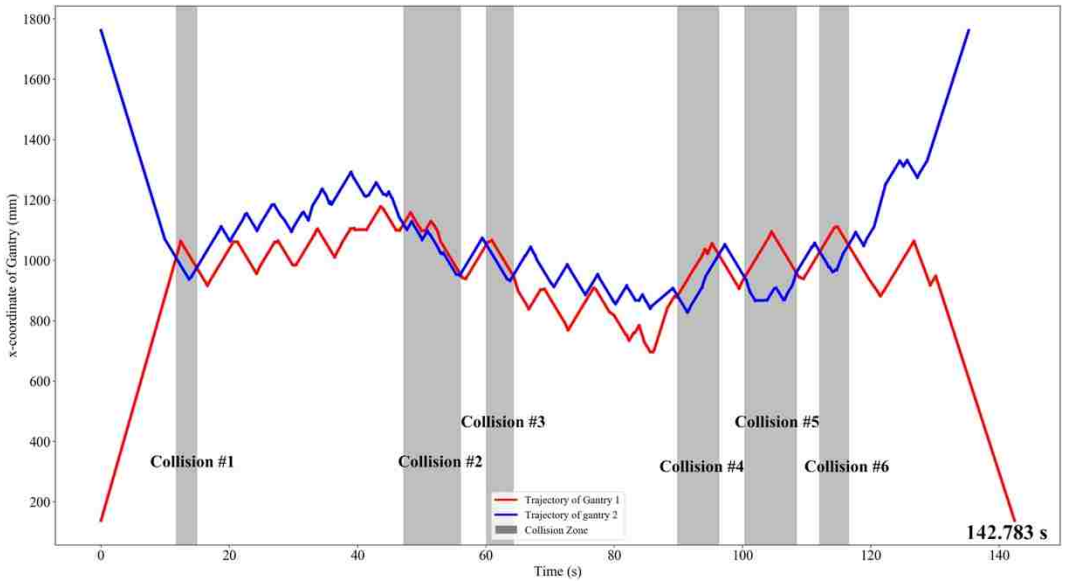


Figure 3.7. Trajectory plot of the toolpath in Figure 3.5, collision checking subroutine identified 6 potential collisions

3.3.2. Collision resolution subroutine

The collision resolution subroutine is developed to tackle the collision by identifying an opportunity for adding a delay and determining the minimum amount of time required to resolve the collision. There are two scenarios that the collision can be resolved by either (1) adding delay to the left trajectory, or (2) adding delay to the right trajectory.

Let P_l and P_r represent the set of vertices of the left and right trajectories associated with the collision, respectively. CI_l and CI_r denote the set of vertex indices of P_l and P_r, respectively.

$$P_l = \{p_{l_i} \mid p_{l_i} = (t_{l_i}, x_{l_i})\ \forall i \in CI_l\}$$

$$P_r = \left\{p_{r_j} \,\middle|\, p_{r_j} = \left(t_{r_j}, x_{r_j}\right)\ \forall j \in CI_r\right\}$$

In order to determine where the delay can be inserted, the subroutine starts by looking at the vertices before the collision and check which one is appropriate to select. In particular, the selected vertex of the left gantry p_{L^*} needs to be less than any vertices in P_r. On the other hand, the selected vertex of the right gantry p_{R^*} needs to be greater than any vertices in P_l.

$$x_{p_{L^*}} \leq x_{r_j} \quad \forall r_j \in P_r \tag{1}$$

$$x_{p_{R^*}} \geq x_{l_i} \quad \forall l_i \in P_l \tag{2}$$

After determining the p_{L^*} and p_{R^*}, the subroutine then calculates the amount of delay needed for each selected vertex. Let d_{L^*} and d_{R^*} represent the amount of delay needed to add to the p_{L^*} and p_{R^*} to resolve the collision, respectively. $t\_dist(p, \overline{uv})$ is the time distance (i.e., amount of delay) from vertex p to the segment $\overline{uv}$.

$$D_1 = \{t\_dist\left(p_{l_i}, \overline{p_{r_j} p_{r_{j+1}}}\right) \mid \forall\, p_{l_i} \in P_l, \forall p_{r_j} \in P_r\ \} \tag{3}$$

$$d_{L^*} = \max\,(D_1) \tag{4}$$

$$D_2 = \{t\_dist\left(p_{r_j}, \overline{p_{l_i} p_{l_{i+1}}}\right) \mid \forall\, p_{r_j} \in P_r, \forall p_{l_j} \in P_l\ \} \tag{5}$$

$$d_{R^*} = \max\,(D_2) \tag{6}$$

The subroutine then picks the selected vertex that requires the least amount of delay to resolve the collision. A small user-defined epsilon (default was set to 0.2s) is also added to the delay to create a separation in the time axis as shown in the zoom section of Figure 3.8.

$$d = \min(d_{L^*}, d_{R^*}) + \epsilon \tag{5}$$

The following trajectory segments after the selected vertex are adjusted by shifting them to the right by d seconds. Since additional collisions might happen down the line after the

adjustment, the collision checking subroutine is needed. These two subroutines are repeated until no further collision remain. This ensures that the result is collision-free. All the delays inserted in this CCR process are recorded in order to adjust the toolpath appropriately. The CCR is designed to be computationally inexpensive as it is frequently used throughout the TS.

Appendix A provides an example of the collision resolution subroutine for a collision. Figure 3.8 illustrates the final trajectory plot of the toolpaths in Figure 3.5. The infill makespan of the solution, in this case, is 166.357 seconds.

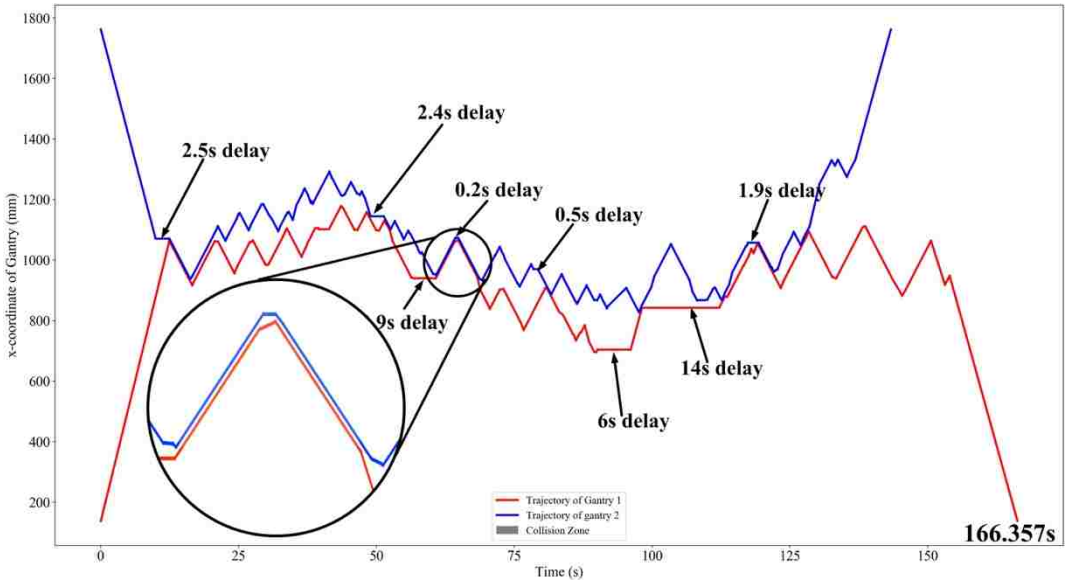


Figure 3.8. Trajectory plot result from the CCR. Several delays were added to solve the 6 collisions in Figure 3.6

3.4. Closed-loop control and resynchronization process.

Many stepper-driven FFF 3D printers operate in open-loop control mode where the printers only execute a series of given instructions, G-code commands, from a controller board and can be affected by some disturbances. To ensure the high quality of the 3D printed parts in term of dimensional accuracy, a closed-loop control system is often required. It allows the controller to know the position of the printhead and make compensations if needed. For multi-gantry 3D printers in this research, the closed-loop control system is especially important as it

ensures the gantries to operate at their intended trajectory and to avoid collisions. In this work, the closed-loop control system is developed to keep track of the x-position of the gantries. Rotary encoders are mounted to the stepper motors that move the gantries. An Arduino Mega is used to collect and forward the data from these encoders to the computer. The encoder readings are updated every 100 ms when the process is running.

The Escher-supported system requires each gantry to have a dedicated controller, meaning that each gantry needs a separate G-code file. The computer sends the G-code commands line-by-line to the controller until its queue is full. To generate the G-code file for each gantry, the associated toolpath is extracted and converted to a series of G-code commands. Even though the proposed method utilized the trapezoidal velocity profile to model the trajectory of the gantries, the actual trajectory might be slightly different. To complicate the matter, the calculated trajectory does not account for any disturbances or uncertainties that could affect the movement of the gantries. For example, latency issues from the serial connections could prevent the gantry from reaching its intended location at the computed time, thus rendering the calculated trajectory incorrect. To resolve this issue, a resynchronization process was designed. The process analyzes the data received from the encoders and determines when the two gantries need to resync to ensure these gantries follow their intended trajectories. By manipulating the G-code sending process, the resynch action can be added during the print. Each resynch action comprises of two G-code commands, M400 and G4. In Marline firmware, when the machine's controller receives the M400 command, it will stop accepting new commands and wait until all the moves in the queue are finished. It is used as a way to clear the command queue before accepting a new command. G4 command is used to pause the machine, the gantry in this case, for a specific amount of time.

The CCR ensures that the trajectories of the gantries do not overlap; this means that there is always a safe margin between them. When the observed separation between two gantries is smaller than the predefined safety margin, the process lets the gantries resync before continuing. It is worth noting that there is a time difference between the time the resync action is added and when it is actually executed. Four separate Python scripts are created to be used for: (1) sending G-code commands to the left gantry, (2) sending G-code commands to the right gantry, (3) reading the data from encoders and identifying opportunity for adding resync action, and (4) keeping track of the state of completion of the resync action for both gantries. In the developed process, the state in the fourth script prevents the gantries from accepting new commands until both of their queues are cleared, which is different from the traditional M400 command. Upon the completion of each resync action, a small delay, represented in G4 command, is added to account for the difference between the starting time of the next trajectories. Figure 3.9 illustrates the flowchart of the closed-loop control with the resync process for multi-gantry 3D printing. In order for the resync process to be successfully implemented, the aforementioned Python scripts need to run concurrently. The multiprocessing package in Python allows modern computers to accomplish such a requirement.

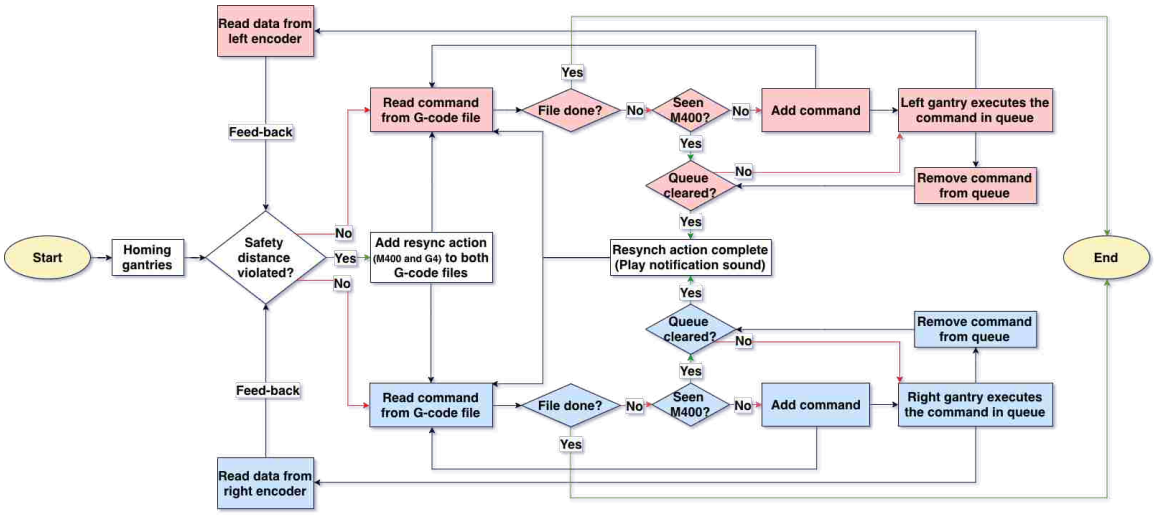


Figure 3.9. Flowchart of closed-loop control with the resynchronization process built-in

3.5. Experimentation

The purpose of the simulation experimentation is to compare the performance of the proposed TS-CCR method with the OSA. This is done on a selected layer of four different 3D CAD objects, each with different layer complexity. These layers' geometries represent some of the practical uses that would benefit from multi-gantry 3D printers as the size of these layers are relatively large, thus requires significant printing time to complete. The physical simulation includes two main objectives. The first objective is to verify that the optimized toolpath is collision-free when implementing it in a custom system that mimics the two-gantry system. The second objective is to validate the print using the proposed method to show the mechanical properties of the finished objects are at least on par when compared to the OSA.

3.5.1. Simulation setup

For each object, a layer of 0.3 mm is sliced with three different approaches and compare their makespans. The makespans from these approaches are:

1) The theoretical minimum makespan is calculated by dividing the makespan of a single printhead printing by two.
2) The OSA makespan is calculated by slicing the layer using Netfabb Multi-gantry FFF Engine plugin.
3) The TS-CCR makespan is calculated by adding infill makespan from the proposed TS-CCR to the makespan of printing the perimeters using one printhead.

The selected layers from 3D CAD objects are shown below:

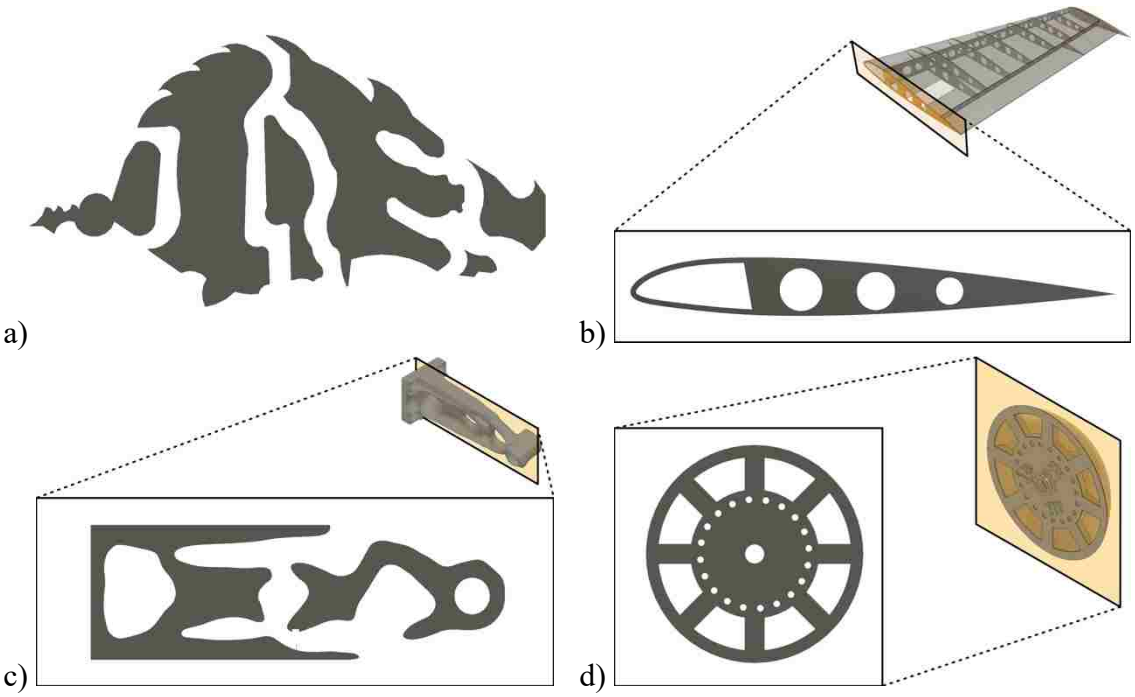


Figure 3.10. a) layer from IE Hog; b) layer from airfoil frame [21]; c) layer from topology optimized bracket [22]; d) layer from rotor hub wind turbine nose [23] .

Titan Cronus multi-gantry printer profile in the Netfabb software was selected as the default value for the printed build volume, 1900 mm x 750 mm x 450 mm (length, width, height). The gantry width of the printer was set to 126 mm with 150 mm of the safety distance. The printing speed was set to 50 mm/s, the rapid travel speed was set to 80 mm/s. The acceleration, deceleration and jerk were set to 2000 mm/s^2, 2000 mm/s^2, and 8 mm/s, respectively. The infill percentage was set to 30%.

For the TS heuristic, the size of candidate list was set to 10. The tabu list size for each operator was chosen as 5. The candidate list size was set at 10. The probability for global swap, local swap, and rebalancing operators were initialized to 0.7, 0.2, 0.1, respectively. The elapsed time was limited to 7 minutes. If the TS could not find solution with over 2% improvement in 3 minutes, the algorithm terminates and returns the best-known solution.

The proposed TS-CCR, simulations, and the physical implementation of the closed-loop control were developed in Python and running on a computer with a 2.6 GHz Core i7 6-core processor and 16 GB RAM.

3.5.2. Custom multi-gantry printer's setup

Since multi-gantry FFF technology is relatively new, the availability of the printers that are designed to run this technology is still limited. To physically verify that the toolpath from the proposed method is collision-free, a custom setup comprised of two Makergear printers was designed as shown in Figure 3.11. The idea of this creation came from the earliest version of the Project Escher printer, where the developers connected the bed of two light single-printhead Printrbot printers and treated each printer as an independent gantry.

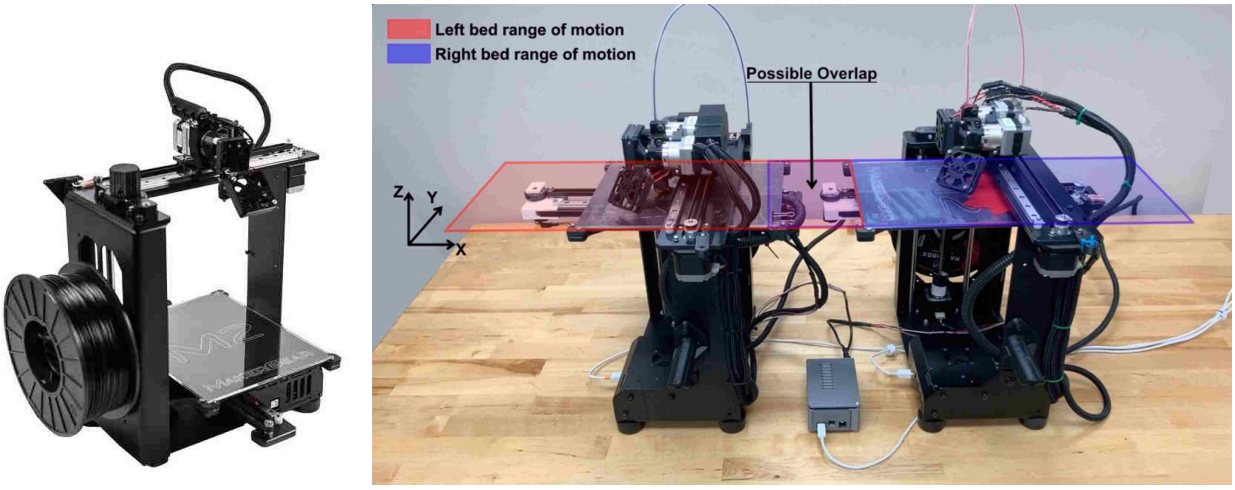


Figure 3.11. a) standalone Makergear M2 3D printer; b) the custom setup comprised of two Makergear printers to test the performance of the closed-loop control.

However, the available Makergear printers are heavy due to the metal construction and would require intensive modification to achieve the similar kinematic configuration of the multi-gantry printers. With the main objective of verifying a collision-free toolpath, the custom setup instead allows the printers' heated beds to collide when each printer is printing its assigned toolpath. In this configuration, the y-direction of the original printer becomes the x-direction in

the custom setup. Since the proposed TS-CCR is designed for multi-gantry printers, before generating the G-code files, the toolpaths is manipulated to fit the custom setup. In particular, the program transforms the toolpaths 90 degrees in the z-direction. The left toolpath is assigned to the right printer and vice versa. In the custom setup, as the left printer begins to print the left area of the layer, the heated bed moves to the right. Similarly, as the right printer begin to print the right area of the layer, the heated bed moves to the left. Thus, it creates a possibility for collision. Collision-free toolpaths mean that when the two printers execute them, their heated beds should not touch each other, and the safety margin between them should always be maintained. As mentioned in the previous section, the commands were sent to the printers concurrently to allow them to work together. The layer from "IE Hog" was selected to run the test. Since the custom setup is smaller than the actual multi-gantry machine. The built platform area was set to 440 x 200 mm. The safety margin was set to 50 mm.

3.5.3. Tensile specimen design

In order to fit into the Universal Testing Machine, small specimens are designed and printed with a custom profile for each printing approach. Instead of using two printheads to concurrently print different portions of the same specimen, one printhead was utilized to print these portions in sequential order. Each specimen comprises of two parts: 3D printed part and four steel tabs, as shown in Figure 3.12d. The 3D printed parts were designed as a rectangular block of size 185 mm x 19 mm x 3.3 mm (length, width, height). These parts are printed with Polylactic Acid (PLA) and the steel tabs were glued to the printed parts using J-B Weld plastic bonder. The layer view of the printed part can be seen in Figure 3.12 a and b. The OSA divided each layer into 2 sub-regions with the seam parallel to the y-axis (Figure 3.12c). This created a

notable seam in the printed parts where each sub-region required its own perimeter shells.

Process parameters used to print specimens are listed in Table 3.1.

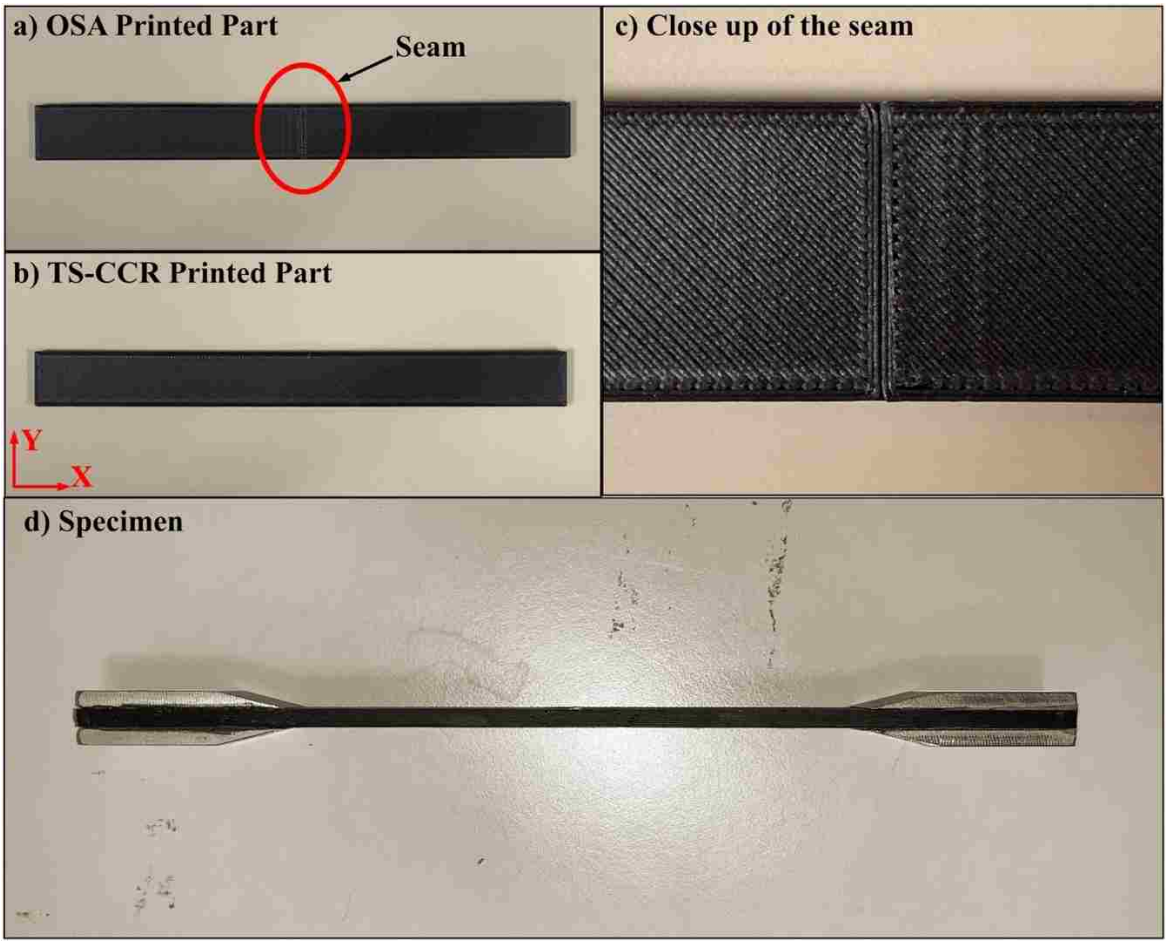


Figure 3.12. a, b) printed part of the specimens using two different approaches; c) close up shot of the seam; and d) complete specimen.

Table 3.1. Process Parameters for 3D printed parts

Process parameters for 3D printed parts (10 specimens)	
Nozzle diameter	0.35 mm
Layer thickness	0.3 mm
Number of perimeter shells	3
Default printing speed	50 mm/sec
Raster angle offsets	45 and -45 degrees
Infill percentage	100%
Extrusion temperature	210°C

3.5.4. Tensile testing procedures

The ASTM D5083 standard was used to measure tensile properties of reinforced thermosetting plastics. Tensile testing measurement procedures were conducted on a MTS testing machine with mechanical wedge grips attached as shown in Figure 3.13. The values of stroke and load data (in pounds) for each specimen were recorded into a csv file for data processing. Stroke is the movement of the piston in inches. Engineering stress, engineering strain, and ultimate tensile strength were chosen to compare the mechanical properties of the two toolpath planning approaches.

- Engineering stress: $s = \frac{F}{A_0}$,where F is the applied tensile force (load), A_0 is the cross-sectional area of the specimen before testing.
- Engineering strain: $e = \frac{\Delta L}{L_0}$ where ΔL is the change in length, L_0 is the original length.
- Ultimate tensile strength: UTS is measured as the maximum value of engineering stress.

a) 

b) 

Figure 3.13. a) MTS tensile testing machine; b) mechanical wedge grips

4. Results

4.1. Simulation results

As mentioned in section 3, three types of makespans were calculated to compare the performance of the proposed TS-CCR method and OSA. They are (1) theoretical minimum makespan, (2) the OSA makespan, and (3) the proposed method makespan. Figure 4.1 illustrates the comparison chart between the three makespans. In all four selected layers, the proposed methodology produces solutions with smaller makespans than the OSA and helps bring the overall makespan closer to the theoretical minimum. The percentage improvement varies depending on the complexity of the layer. The proposed method reduces the makespan by 15.14% on the simple Airfoil frame layer, while the improvement reduces to 7.72% for the "IE Hog."

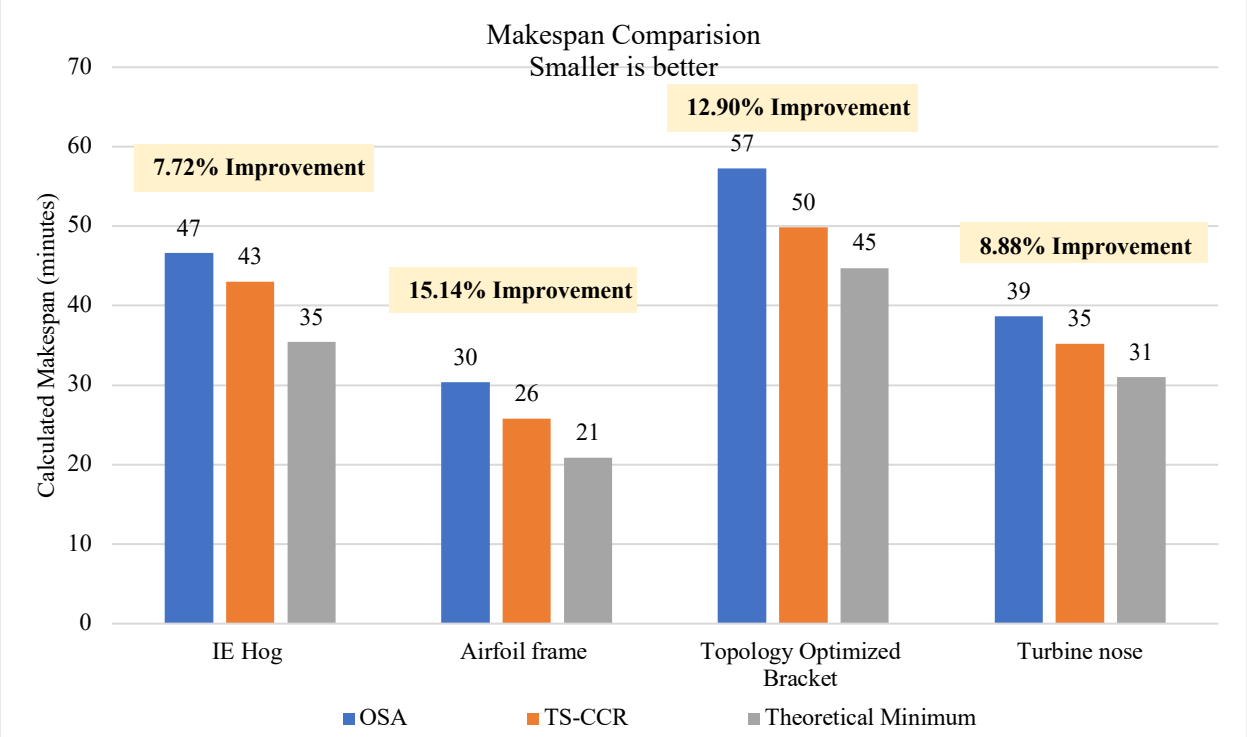


Figure 4.1. Makespan comparison of four selected layers at 30% infill, where the proposed TS-CCR can yield solutions with shorter makespan than solution obtained from OSA.

4.2. Physical experiment results

While it is important for the TS-CCR to reduce the makespan, it is also important that the method does not negatively impact the mechanical strength of the printed products. This section provides the result from the tensile test on the specimens that were sliced with the two toolpath planning approaches and were printed on a single gantry machine. It is also crucial that the toolpaths from the proposed method can be implemented on the machine. This was validated by running the TS-CCR toolpaths on the aforementioned custom setup machine with the closed-loop control and resynchronization process running on the background.

4.2.1. Results from tensile test

Figure 4.2 a and b shows diagrams of stress vs. strain for specimens printed with the OSA and the proposed TS-CCR methods, respectively. On average, the UTS of the TS-CCR specimen (5991.465 psi) was found to be stronger in comparison to the UTS of the OSA specimen (4443.416 psi). The OSA specimens fractured immediately after the yield point at the seam while the TS-CCR specimens fractured after undergoing some plastic deformation. This means the printed parts using the proposed method have a higher ductility/toughness. **Table 4.1** shows that the UTS standard deviation of the TS-CCR specimens is smaller than the other. A two-sample t-test was conducted to confirm that the two UTS means are different. The P-value of 0.0128 indicates that the two UTS results are statistically significant at the 5% significant level.

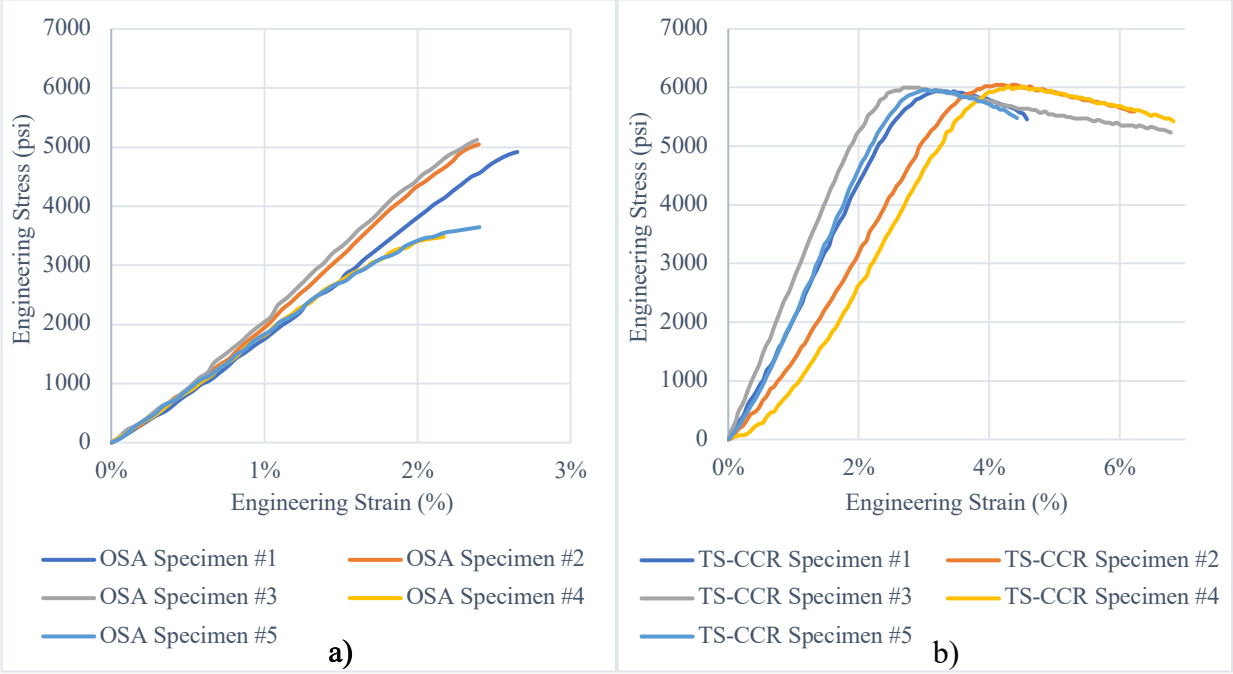


Figure 4.2. Stress-strain plot for specimens printed with a) OSA method, b) TS-CCR method.

Table 4.1. Results from tensile testing.

	UTS (psi)					Average	STD
OSA	4918.398	5047.473	5122.672	3483.369	3645.169	4443.416	807.894
TS-CCR	5935.702	6044.905	5998.208	6020.388	5958.124	5991.465	44.606

Figure 4.3 illustrates the break behavior of the specimen printed with TS-CCR and OSA. OSA's specimens show a brittle fracture. The fracture can be observed at the seam, which is parallel to the y-direction of the built. The void between the perimeters causes the break of one layer and initiates the total break of the entire specimen.

Figure 4.3. Fractures of specimens printed by two different methods.

4.2.2. TS-CCR and closed-loop control validation

The “IE Hog” with 100% infill density was chosen to validate the collision-free toolpaths and the performance of the closed-loop control system. Figure 4.4 shows the layers that were printed on the custom-setup machine. Specifically, the red portion was printed by the right gantry, while the black portion was printed by the left gantry. Note that only the infill toolpaths were printed. The speed was reduced to 50% to ensure the layers adhere well to the heated beds.

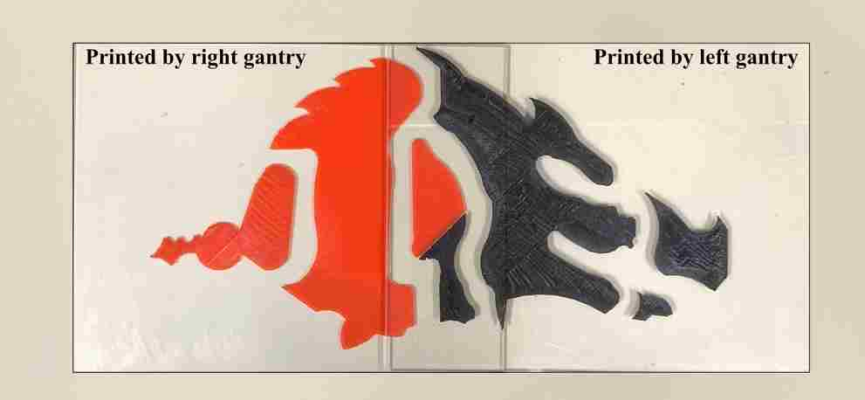


Figure 4.4. Finished layers

Figure 4.5 illustrates the setup of the printers when printing the layer. A safety distance (yellow shaded region) was always maintained by the closed-loop control process. Data from encoders were also recorded for comparison (Figure 4.6b). The observed trajectory plot is comprised of two parts: (1) the trajectories associated with the calibration of the machine, and (2) trajectories associated with the actual printing. In the custom-setup machine, the time spent preparing each printer/gantry can differ up to a few minutes (i.e., heating the bed from room temperature to 70^{o}C). One resync action was added at the beginning of the actual printing that allows both gantries to begin their job at the same time. However, this action was not counted in the comparison. The encoders began to read the position when the gantries start the homing process. In total, two resync actions were introduced during the printing process to allow each gantry to follow its intended trajectory. This means that the closed-loop control and

resynchronization process were able to identify and resolve the minor differences in the calculated and actual trajectories. The actual printing with two resync actions took 1,346 seconds to complete, 3 seconds more than the calculated makespan (Figure 4.6).

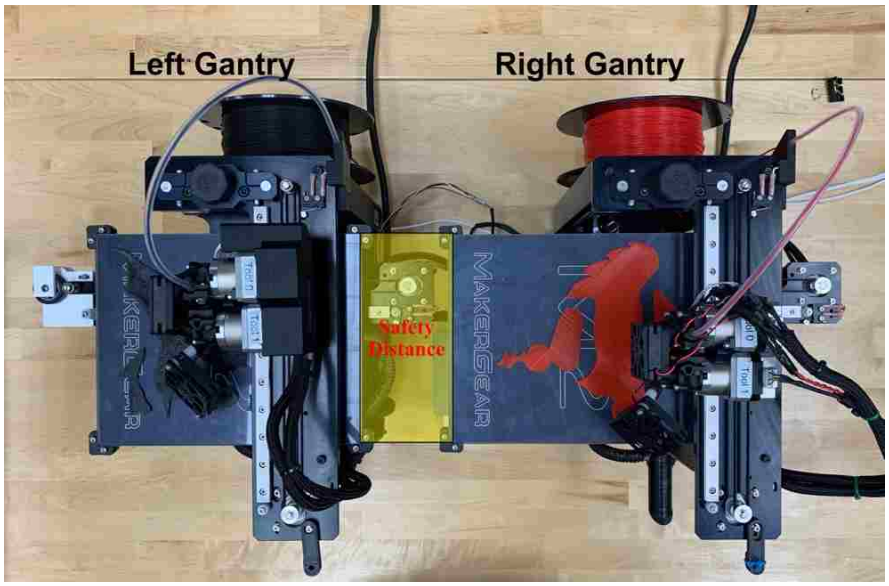


Figure 4.5. The custom setup consists of two printers, each printing the assigned part. While printing, the two heated bed never touching each other. The safety distance (yellow shaded region) were violated twice result in 2 resync actions

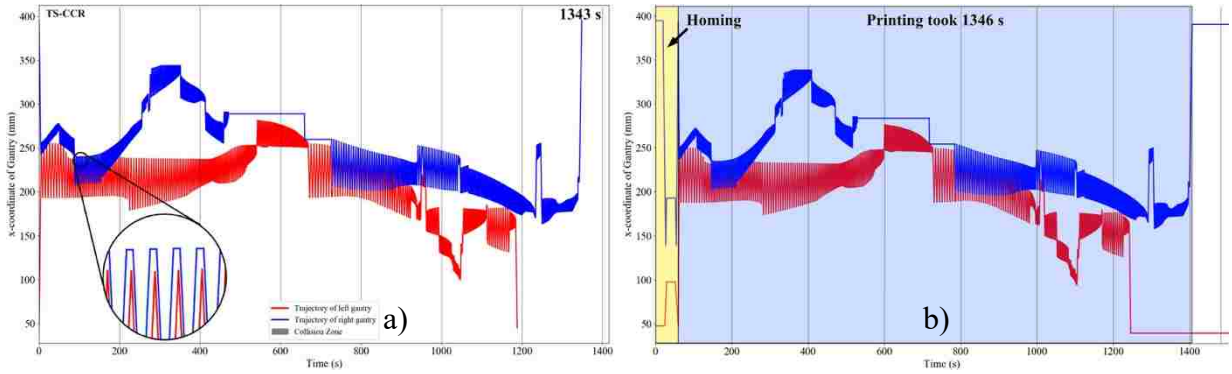


Figure 4.6. a) trajectory plot generated from the TS-CCR for "IE hog" in **Figure 4.4** (zoomed section shows that the two trajectories do not overlap); b) trajectory plot generated from the encoders, two resync actions allow the gantries synchronized throughout the print. These two resync actions result in 3 seconds different between two trajectory plots (note that the homing duration was excluded when calculating the actual makespan)

5. Discussions

5.1. Performance of TS-CCR

As shown in section 4, the TS-CCR yields better solutions in term of makespan when compared with those obtained from OSA. The proposed method can intelligently, without the help of the user, assign the raster segments to each gantry in a way that does not create a collision. There are primary two factors that can affect the performance of the proposed method, namely (1) layer features, and (2) process parameters.

Layer complexity plays an important role in determining the performance of the TS-CCR. Because the proposed method was designed only to optimize the infill aspect of the layer, the perimeters are printed using only one gantry. This means for any complex layer that requires significant time to print the perimeters, it would offset the improvement that the proposed method could achieve. For example, “IE Hog” layer contains several irregular shapes that require the gantry to spend a long time to print these perimeters, thus increasing the overall makespan. Specifically, 5 minutes and 24 seconds were required to print the perimeters, which was 12.55% of the overall makespan. Future research on a strategy that allows multiple gantries to simultaneously print the perimeters could be implemented to reduce the overall makespan even further.

As mentioned in the method section, the gantry movements in the x-direction must be carefully planned to avoid collisions. Thus, the x-dimension of the 3D model is the most critical factor that determines whether that model can be printed on the multi-gantry 3D printer using the proposed method. Two features of the layer that affect the performance of the proposed method are: (1) the aspect ratio between the x- and y- dimension of the layer, and (2) the ratio between the x-dimension of the layer and the width of the gantry. The proposed method performs well

when these two ratios are high as shown in “airfoil frame” layer. As these ratios decrease, the chance for the gantries to collide is increased. This means the CCR is expected to introduce more delays to resolve all the collisions, thus reducing the effectiveness of the method. When the value of the ratio between the x-dimension and the gantry’s width is too low, the gantries have little room to operate. The OSA might fail to produce any result while the TS-CCR might produce the result with larger makespan than the single printhead printing approach. In this case, the whole layer will be printed by one gantry. These two ratios can be adjusted by rotating the 3D model on the z-axis. Thus, the orientation and the size of the 3D model must be analyzed before applying the proposed method.

Safety distance was determined by the user. There is currently no method to define the optimal safety distance value, but it can be done by the user expertise or by trial-and-error approach. Like the gantry width, the safety distance limits how much each gantry can move without registering a collision. A good starting value can be set to approximate the gantry width (i.e., 150 mm for the safety distance vs. 126 mm for the gantry width). Note that the gantry width value is fixed for a machine while the safety distance can be adjusted. Increasing this value requires the TS-CCR more time to run to find good solutions. A fine-tuned machine might not need this value to be high since the user can have confidence that the gantries do not crash into each other.

In TS, each solution is evaluated by the CCR. Although the CCR can evaluate the solution relatively quickly (~300 ms for a given solution), the large number of solutions results in significant computation time. Thus, the size of the candidate list was found to be the most important TS parameter that affects the computational time of the proposed method. Depending on the different layers’ geometries, the size of the candidate list required to be adjusted. The

initial probability of each operator was also found to be an important parameter in TS. As the TS is executed, the need for global swap is diminished in favor of local swap since the number of raster segments begins to allocate well to each gantry. These values determine how fast the TS is switching from choosing the global swap to the local swap in each iteration. The values listed in the above section were found to be appropriate for the chosen layers.

Another aspect of the process parameters can be expressed as the infill percentage. As the infill percentage increases, the number of raster segments increases. The TS-CCR is expected to require additional time to find a good solution. The computing time to optimize each layer is significantly smaller than its makespan. For example, the TS-CCR on each layer was limited to 7 minutes while the makespan is 26 minutes for "airfoil frame" layer and 50 minutes for the "bracket" layer. However, it is computationally expensive to optimize every layer before sending them to the machine. One possible way to mitigate the speed limitation is allowing the machine to execute the current layer n while spending time performing TS-CCR on the next layer $n + 1$. The advantage of this is that allowing the TS-CCR to run for a longer duration on each layer can potentially generate better solutions. Another improvement to the TS-CCR can be made by improving the efficiency of the collision resolution subroutine. The current subroutine resolves the collision by adding a simple delay to one of the gantries. This means the gantry stops and waits until the toolpath is clear to execute. However, instead of staying at one place while waiting (i.e., no change in the gantry's x-coordinate in the trajectory plot), the gantry could potentially move itself out of the way and let another gantry continue to print. This provides extra flexibility when dealing with collisions, thus reducing the number and duration of the delays needed. New collision resolution subroutine will need to be developed to account for the new change.

The methodology proposed in this paper is flexible and can be useful in other configurations beyond multi-gantry. Extending the current method to cover more than two gantries could be the next step to leverage the full potential of multi-gantry printing (e.g., the Titan Cronus printer is equipped with five gantries). The CCR on different trajectory plots of different axes can be incorporated in the TS to find the optimal path for mobile robots or arm-based systems.

5.2. Physical test

The closed-loop control system and resynchronization process were developed to adjust the input of G-code commands to reduce the error between the actual trajectories and the desired ones. Also, the closed-loop control system in this research was desired to only keep track of the movements in the x-direction as a function of time. This means the system did not consider the missed step issue. For example, if the one stepper motor loses steps during a print, then all the layers following it get misaligned, and this results in a failed print. A system that allows the firmware of each gantry to adjust the printhead target position when the motor loses steps has not been investigated. Even though this issue is uncommon, it is unrecoverable under the developed system. The generated toolpaths do not guarantee to be collision-free anymore. To ensure that two gantries do not collide when this problem arises, the M112 command can be utilized to immediately shut down and prevent damaging the machine.

As shown in the result section, all the OSA specimens broke at the seam due to the initial separation between the different sub-regions' perimeters at one layer. The layer sliced by OSA has various seams, as shown in **Figure 1.1**. This means there is more chance for the layer to fail under stress, thus compromising the mechanical properties of the whole printed part. Because the proposed method utilizes one gantry to print the perimeters resulting in no seam, the printed part achieves better mechanical properties.

6. Conclusions

A new toolpath planning methodology TS-CCR, has been developed to generate the collision-toolpath for two-gantry FFF printer. The TS-CCR have been shown to provide solutions with shorter makespan than the available approach, OSA, while achieving good mechanical properties. The toolpaths of one selected layer were printed on a custom setup machine that mimics the multi-gantry printer. It helped validate that the TS-CCR is capable of producing collision-free toolpaths. A closed-loop control system and resynchronization process were developed to monitor the execution of the TS-CCR toolpaths. The developed control system demonstrated the ability to detect the differences between the calculated and observed trajectory plots and use the resync action to correct them. While the two-gantry FFF printer was chosen to develop the method, the proposed method can be expanded to cover more gantries, thus reducing the fabrication time even further.

7. References:

[1] J. Lee, J. An, and C. Chua, "Fundamentals and applications of 3D printing for novel materials," vol. 7, ed: Applied Materials Today, 2017, pp. 120-133.

[2] N. Li, Y. Li, and S. Liu, "Rapid prototyping of continuous carbon fiber reinforced polylactic acid composites by 3D printing.," vol. 238, ed: Journal of Materials Processing Technology, 2016, pp. 218-225.

[3] B. G. Compton and J. A. Lewis, "3D-Printing of Lightweight Cellular Composites," *Advanced Materials,* vol. 26, no. 34, pp. 5930-5935, 2014

[4] N. Volpato, R. Nakashima, L. Galvao, A. Barboza, P. Benevides, and L. Nunes, "Reducing repositioning distances in fused deposition-based processes using optimization algorithms," in *6th International Conference on Advanced Research in Virtual and Rapid Prototyping*, Leiria, Portugal, 2014, pp. 417-422.

[5] N. Ganganath, C. Cheng, K. Fok, and C. K. Tse, "Trajectory planning for 3D printing: A revisit to traveling salesman problem," in *2016 2nd International Conference on Control, Automation and Robotics (ICCAR)*, 2016, pp. 287-290.

[6] K.-Y. Fok, C.-T. Cheng, N. Ganganath, H. Ho-Ching Iu, and C. Tse, "Accelerating 3D Printing Process Using an Extended Ant Colony Optimization Algorithm," presented at the 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 2018.

[7] P. K. Wah, K. G. Murty, A. Joneja, and L. C. Chiu, "Tool path optimization in layered manufacturing," *IIE Transactions,* vol. 34, no. 4, pp. 335-347, 2002.

[8] M. Wojcik, L. Koszalka, I. Pozniak-Koszalka, and A. Kasprzak, "MZZ-GA algorithm for solving path optimization in 3D printing," presented at the The Tenth International Conference on Systems (ICONS 2015), 2015.

[9] Y. Jin, H. A. Pierson, and H. Liao, "Toolpath allocation and scheduling for concurrent fused filament fabrication with multiple extruders," *IISE Transactions,* vol. 51, no. 2, pp. 192-208, 2019.

[10] Y. Jin, H. A. Pierson, and H. Liao, "Concurrent fused filament fabrication with multiple extruders," in *IIE Annual Conference*, 2017: Institute of Industrial Engineers (IISE), pp. 940-945.

[11] X. Zhang *et al.*, *Large-scale 3D printing by a team of mobile robots*. 2018, pp. 98-106.

[12] S. H. Choi and W. K. Zhu, "A dynamic priority-based approach to concurrent toolpath planning for multi-material layered manufacturing," *Computer-Aided Design,* vol. 42, no. 12, pp. 1095-1107, 2010.

[13] I. Wagner, Y. Altshuler, V. Yanovsky, and A. Bruckstein, *Cooperative Cleaners: A Study in Ant Robotics*. 2008, pp. 127-151.

[14] L. Tsai-Yen and J. Latombe, "On-line manipulation planning for two robot arms in a dynamic environment," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*,1995, vol. 1, pp. 1048-1055 vol.1.

[15] K.-H. Lee and J.-H. Kim, "Multi-robot cooperation-based mobile printer system," *Robotics and Autonomous Systems,* vol. 54, no. 3, pp. 193-204, 2006.

[16] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics and Automation,* vol. 14, no. 6, pp. 912-925, 1998.

[17] M. Peasgood, C. M. Clark, and J. McPhee, "A Complete and Scalable Strategy for Coordinating Multiple Robots Within Roadmaps," *IEEE Transactions on Robotics,* vol. 24, no. 2, pp. 283-292, 2008.

[18] E. Todt, G. Rausch, and R. Suarez, "Analysis and classification of multiple robot coordination methods," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 24-28 April 2000 2000, vol. 4, pp. 3158-3163 vol.4.

[19] F. Glover, "Tabu search," in *Encyclopedia of Operations Research and Management Science*. Boston, MA: Springer US, 2001, pp. 821-827.

[20] F. Antonio, "IV.6 - FASTER LINE SEGMENT INTERSECTION," in *Graphics Gems III (IBM Version)*, D. Kirk Ed. San Francisco: Morgan Kaufmann, 1992, pp. 199-202.

[21] R. Kumar. "Wing - 23012 Aerofoil Section." https://grabcad.com/library/wing-23012-aerofoil-section#_=_ (accessed 2019).

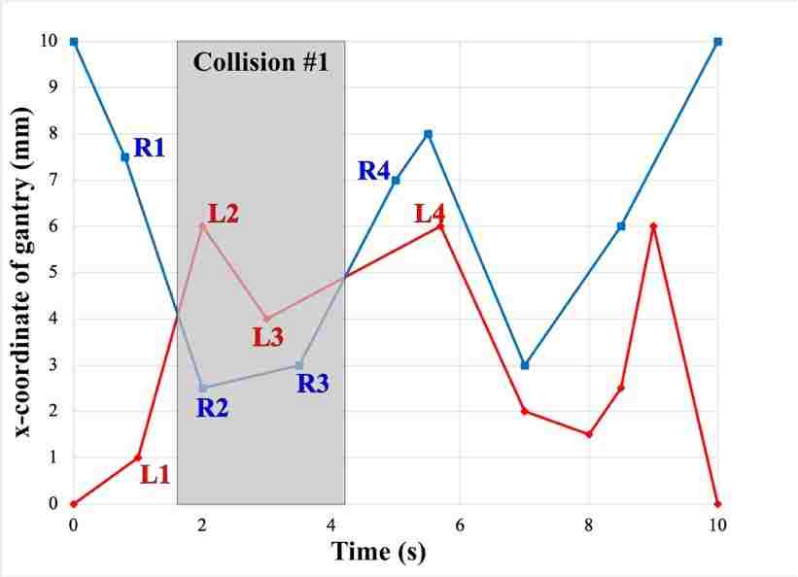
[22] P. Technology. "Bracket topology optimization 2." https://grabcad.com/library/bracket-topology-optimization-2-1 (accessed 2019).

[23] B. Salaets. "Rotor hub wind turbine." https://grabcad.com/library/rotor-hub-wind-turbine-1 (accessed 2019).

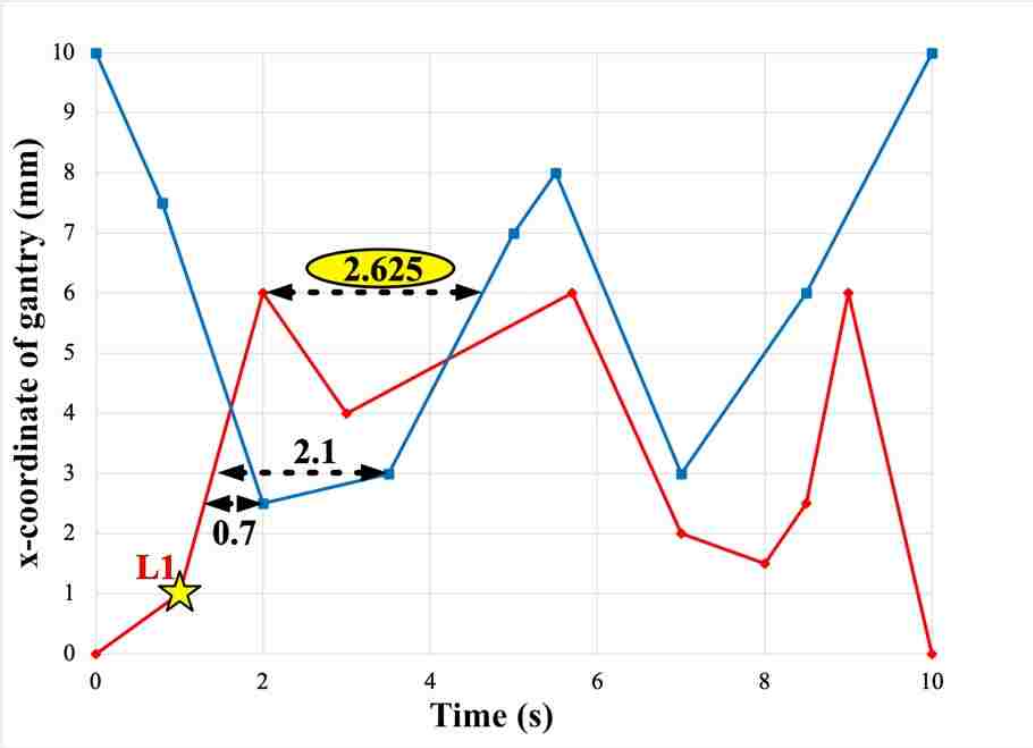
8. Appendix

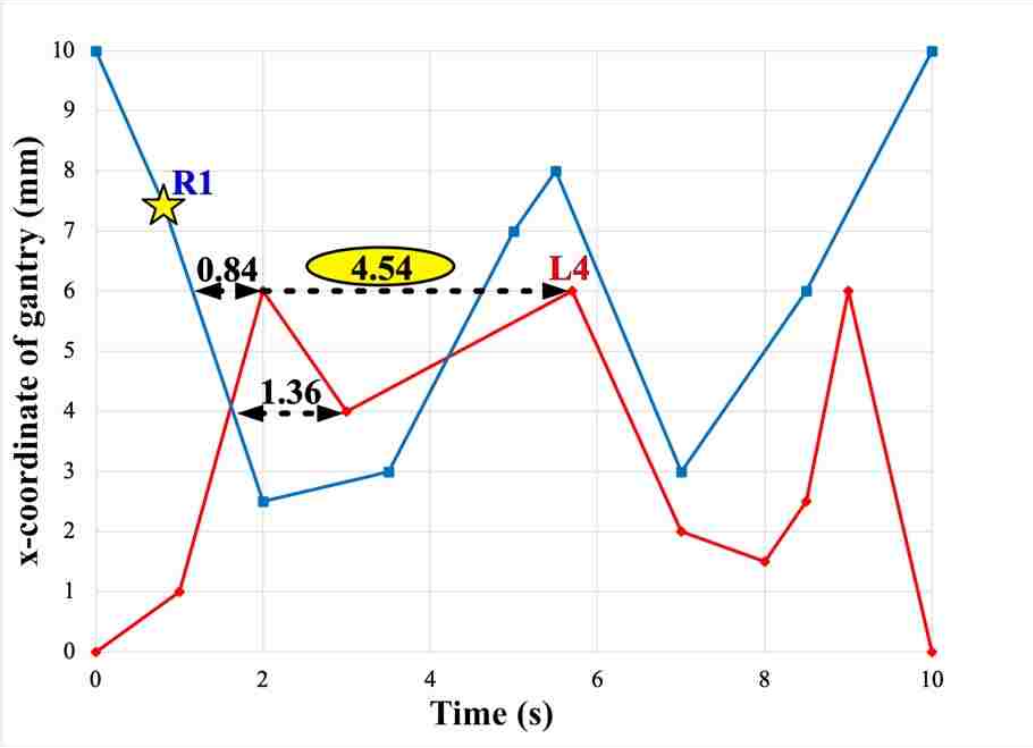
8.1. Collision resolution subroutine illustration.

Figures below demonstrate the logic of the collision resolution subroutine to solve a single collision. Given a simple example below where one collision is detected, all vertices associated with the collision are recorded. L1 and R1 are selected as p_{L^*} and p_{R^*}, respectively.



After identifying the p_{L^*} and p_{R^*}, the subroutine identifies the amount of delay required to resolve the collision at each selected vertex. In this example, the $d_{L^*} = \max(0.7, 2.1, 2.625) = 2.625$ seconds is selected so if it is added to the *L1*, the collision can be resolved. Similarly, $d_{R^*} = \max(0.84, 4.54, 1.36) = 4.54$ seconds is selected.





The subroutine selects $d = \min(2.265, 4.54) = 2.625$ s delay to resolve the collision. In addition, 0.2 seconds is added which bring the total delay to 2.825 s. All trajectories after p_{L^*} or L_1 are adjusted accordingly. After the adjustment, the collision checking subrountine is used to identify either (1) additional collision is created, or (2) no collision is remaining.