Improved Collision-Free Multi-Axis Tool-Path for Additive Manufacturing

Rahnuma Islam Nishat[1] , Yeganeh Bahoo[2] , Konstantinos Georgiou[2] Robert Hedrick[3] , R. Jill Urbanic[4]

[1]University of British Columbia, Canada, rahnuma.nishat@ubc.ca
[2]Toronto Metropolitan University, Canada, {bahoo,konstantinos}@torontomu.ca
[3]CAMufacturing Solutions Inc., Canada, bob.hedrick@camufacturing.com
[3]University of Windsor, Canada, jurbanic@uwindsor.ca

Corresponding author: Yeganeh Bahoo, bahoo@torontomu.ca

Abstract. The focus of this work is on Additive Manufacturing (AM) where the material is deposited layer by layer. In AM, a generated tool-path dictates how and in what order the deposition head should move to fabricate the complex components. In bead based deposition AM processes, such as directed energy deposition and multi-axis material extrusion systems, collision avoidance issues can arise. As such, it is crucial that a tool-path does not cause collisions between the deposition head and the rest of the object. In this work, given a tool-path, we modify the path based on an heuristic approach such that the generated tool-path does not collide with the rest of the object if possible. We test our algorithms with industrial standard softwares, APlus and Mastercam. Our experimental results show that the heuristic approaches improve the computational time significantly compared to our previous deterministic algorithm for collision avoidance.

Keywords: Collision Avoidance, Mesh, Additive Manufacturing, Multil-axis Tool-path, Heuristic Methods
DOI: https://doi.org/10.14733/cadaps.2024.1103-1115

1 INTRODUCTION

Additive manufacturing (AM) is a new way of manufacturing in which materials are fabricated layer by layer, increasing the volume of the object being printed over the course of manufacturing. The increase in the volume causes higher chances of collision between the machine parts and the printed material during the fabrication process. In many additive manufacturing system, a planar '$2\frac{1}{2}$D' build strategy is employed where collision avoidance issues do not exist between the object being printed and the AM heat source solution (e.g., a laser or a a material deposition nozzle). However, the directed energy deposition (DED) processes use a heat source and material feeding system mounted on a multi-axis CNC system or a robot to deposit beads side by side to fill a layer. The DED processes are getting popular as they can be used to fabricate parts without support structure, repair damaged parts, or in surface coating, by leveraging non-planar slicing and multi-axis tool-paths. This multi-axis solution introduces potential collision issues.

The state-of-the-art algorithms for designing tool-paths for multi-axis additive manufacturing do not consider the above collision issues and thus, have no techniques to avoid such collisions. However, such collisions does not only affect the printing process and the quality of the printing, but can also incur major damages to the machines themselves. Therefore, addressing this issues is of utmost importance and a matter of urgency . We need to put special consideration in the design of the *tool-path*, which is the path on which the *deposition head* of the machine moves on during the manufacturing process. A *tool-path* is a path defined by a sequence of points in 3D along with *tool vector* for each tool-path point which is a vector representing the direction of the deposition head at that point, and difference between tool vectors of two consecutive points of the tool-path does not exceed some given threshold.

Although several approaches for detecting and avoiding collisions have been investigated for CNC machining (see [15] for a survey), it is a relatively new research area for AM processes. The AM problem space has unique challenges to overcome. First of all, the deposition head may not be rotationally symmetric as shown in Figure 1(a)–(b), where the cutting tools in machining are symmetric. Secondly, as mentioned above, AM

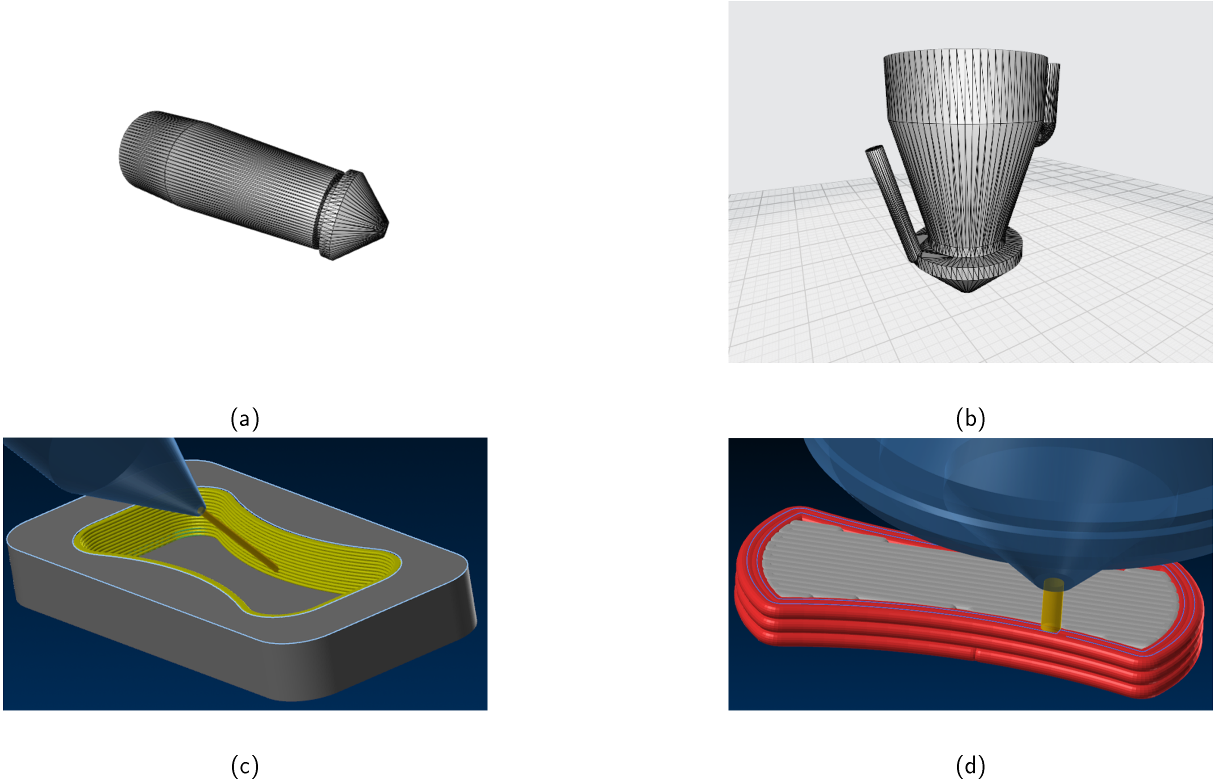


Figure 1: (a) Rotationally symmetric holder, and (b) rotationally asymmetric holder; (c) printing inside a cavity where the deposition head may collide with the object; (d) printing boundary of an object.

process adds material to the workspace in contrast to subtractive manufacturing or machining where materials are removed over time, thus increasing the probability of collision with the progress of the building process. Finally, the build shape (see Figure 1(c)–(d) for some example) cannot be accurately represented due to the basic imprecision of the DED AM process.

Previously, we gave an algorithm that modifies a given multi-axis AM tool-path to a collision-free tool-path [12]. Using 3D triangle-mesh representations for both the object to be printed and the deposition head, we first applied algorithms from the literature to detect collision between those two meshes. If collision is detected, we built a *configuration graph* considering multiple tool vectors for each tool-path point as vertices and computed a collision-free tool-path, if it exists, using graph algorithms to compute a path in the configuration graph containing a collision-free tool vector for each tool-path point. Although the algorithm showed promising results, the time required by the algorithm for complicated and big tool-paths was not up to the standard in the AM industry. In this paper, we propose heuristic based approaches to improve the performance of the previous algorithm significantly.

The rest of the paper is organized as follows. We discuss some related work in the next section and also give an overview of our previous algorithm from [12]. We then describe our proposed heuristic approach. Finally, we conclude the paper with a discussion on future direction of research in avoiding collision in multi-axis additive manufacturing tool-path generation.

2 RELATED WORK

Collision avoidance in CNC machining has been studied for both *local* (i.e., local gouging, rear gouging, etc.) and *global* collisions. Balasubramanium et al. [2] gave a three-step algorithm to generate tool-path using visibility and accessibility-based method. They compute *visibility cones* (a set of angles from which the given tool-path point is visible from an observer outside) for every the tool-path. However, since visibility does not ensure accessibility (i.e., the machining tool cannot access the point without colliding), the second stage checks for accessibility of the tool for each of the angles in the visibility cone; and computes the set of angles which are *valid* or collision-free. Finally, a continuous tool-path is computed from the sets of valid angles for each tool-path point. Balasubramanium et al. [3] presented another algorithm in 2003 using similar approaches to deal with global collision detection. However, their algorithm is computationally expensive as the objects being machined are presented as *clouds of points*.

Wang and Tang [17] proposed an algorithm to identify the set of valid orientations or configurations by inspecting the valid area with all the gouging constraints. But their method had a high space and time complexity as well.Xu et al. [18] applied a similar approach. Li and Zhang [8, 9] used point cloud-based approach, which requires huge computer resources in terms of space and time, similar to the algorithms of Balasubramanium et al.

Aliyu and Al-Sultan [1] gave linear programming algorithm to detect collisions between moving objects, where the objects are represented as polyhedral sets in $\mathbb{R}^2$ and $\mathbb{R}^3$. Tang et al. [14] gave a sweep plane based algorithm to detect global collisions in 5-axis NC machining using the workpiece slices and machine component slices. They also gave an algorithm to avoid collision in 5-axis NC machining [16].

Collision detection and avoidance has attracted attention in additive manufacturing as well. Nishat et al. [11, 12] gave an algorithm to obtain collision-free tool-paths for multi-axis additive manufacturing (see the next section for details on the algorithm). Fang et al. [5] studied collision-free printing of a layer by selecting a 'best' setup orientation, similar to choosing collision-free tool vectors in our algorithm from [12]. Plakhotnik et al. [13] studied collision avoidance by updating the tool vectors in multi-axis additive manufacturing. They penalized the tool vectors as they get further from the position normal to the surface. However, our algorithm [12] penalized bigger differences between tool vectors of adjacent tool-path points. Jiang et al. [7] studied scheduling in path planning to avoid collision in additive manufacturing when multiple deposition heads are employed.

In computational geometry, detecting intersection between geometric objects has been studied for decades. Möller [10] and Held [6] studied detecting intersection between 3D triangles. Guigue and Devillers [4] improved the above algorithms using floating-point calculation.

3 COMPUTING COLLISION-FREE TOOL-PATH

In this section, we give a summary of our previous algorithm [12] to generate collision-free tool-path. We propose algorithms to improve this approach in the next section.

Given a tool-path, which includes a set of continuous positions of the deposition head a long with tool vector of each points (direction of the deposition head), our goal is to find a collision-free tool-path if it exist for this set of points. We present both the surface of the CAD model and the deposition head as triangular mesh. We considered both symmetric and asymmetric deposition heads; see Figure 1(a) and (b), respectively, for symmetric and asymmetric deposition head mesh. An example run of the algorithm is shown in Figure 3, the tool-paths were generated using APlus software on Mastercam platform.

We applied two geometric approaches to detect collisions between the deposition head and the object being printed: 1) Möller's algorithm [10], and 2) a commercial third-party library. The first was based on calculation of collisions between each pairs of triangles of the mesh of the object and the mesh of the deposition head. To speed up the Möller's technique we used the parallel computation. The second approach was based on clash detection between two solids representing the deposition head and the object; multi-threading was used to apply this technique.

If collision was detected, our algorithm modified the given tool-path to a collision-free tool-path. We built a *configuration graph* where each vertex represents a tool vector at a specific tool-path point. A set of *feasible* tool vectors was assigned to each point of the tool-path, where a *feasible* tool vector means that the deposition head does not collide with the object when aligned along this tool vector at that specific tool-path point. To generate feasible tool vectors, we generate unit tool vectors on a unit sphere where the center of the sphere is the tool-path point as shown in Figure 2. Here, θ is the maximum allowable change of angle for the tool

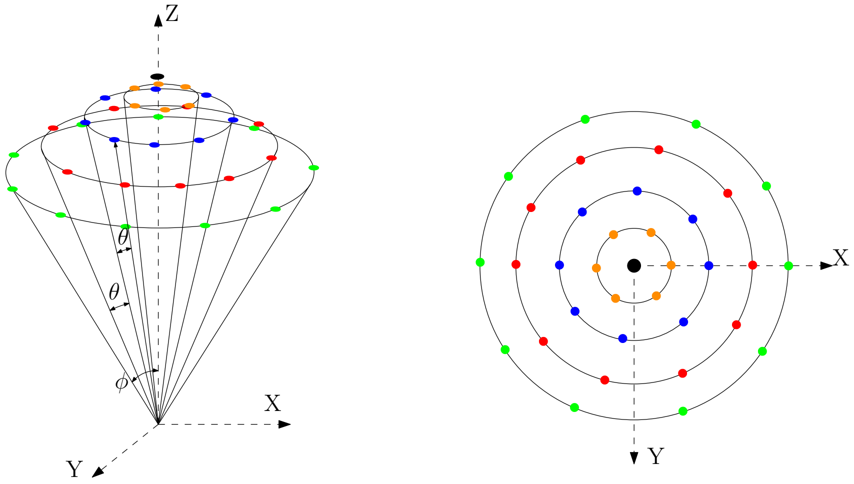
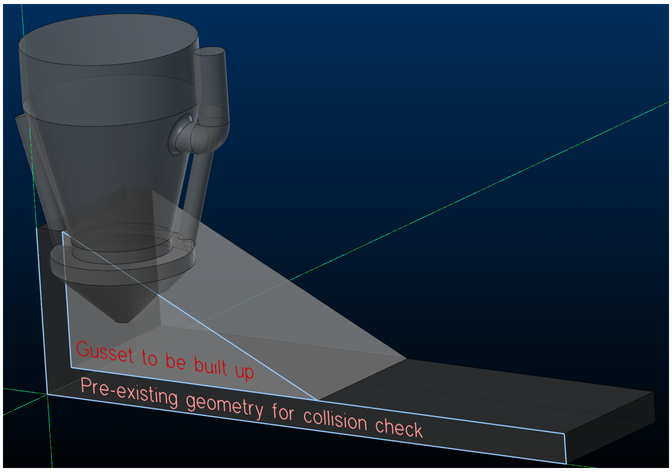


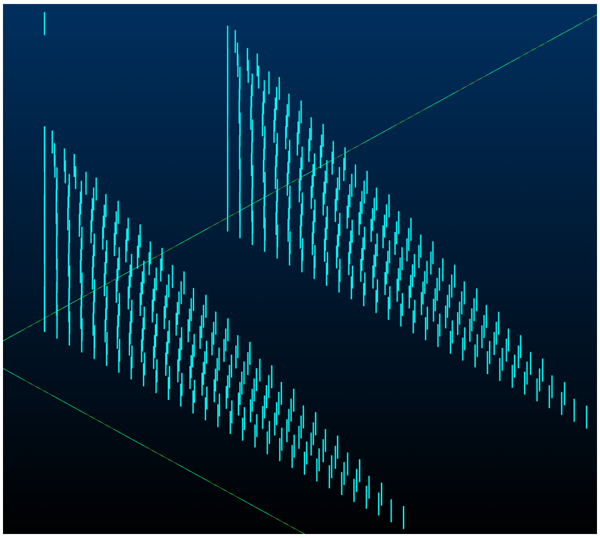
Figure 2: [12] Generating tool vectors on a unit sphere within the given conic volume around a normal along Z-axis; isometric view (left) and top view (right) are shown.

between adjacent tool-path points, and ϕ is the maximum tilt angle for the tool/holder. As the figure shows, the distance between consecutive cycles is θ, as is the distance between two consecutive tool vectors on the same circle.

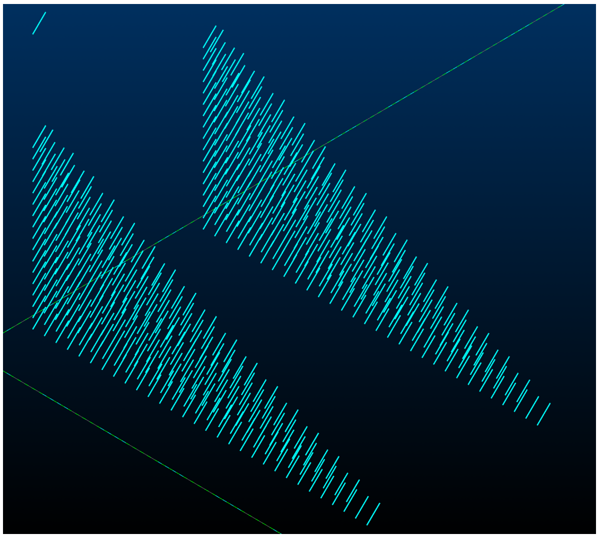
Two vertices will be connected by an *edge* in the configuration graph if and only if their representative tool vectors are for two consecutive points on the tool-path, and the deposition head can change its direction by the difference between the angles of this two tool vector, which is determined based on the mechanical



(a)



(b)



(c)

Figure 3: [12] Generating collision-free tool-path from a given tool-path: (a) tool-path with 566 points creates collision between the printed surface and the asymmetric deposition head; (b) original tool vectors for (a); (c) Modified tool-path that avoids collision.

configuration of the machine. This connection is directional, connecting the vertices of the tool-path position x to y when x appears before y. We created a start point s and connect s to all the vertices allocated to the first point of the tool-path. We also created the vertex t and connected all the vertices associated to the last point of the tool-path to t. After creating the above directed configuration graph, we ran the breadth-first search or Dijkstra's shortest path algorithm to find a path from s and t through all the tool-path points. If such a path exists, that represents a collision-free tool-path for the deposition head. Otherwise, no collision-free tool-path exist for the deposition head to print on the given path. Applying Dijkstra's algorithm gave us the path that allowed minimum change of angles between tool vectors of consecutive tool-path points.

The configuration graph in our algorithm [12] was large even for tool-paths which includes a few hundreds of points. This slowed down the computation of the path from s to t. To improve the time complexity, we have studied different ways of approaches including heuristic approach to improve the time. We will go over the details of our approach in the next section.

4 HEURISTIC APPROACHES

In order to reduce the time we need to calculate the collision-free tool-path, we reduced the size of the produced graph. Such reduction can be done by combination of the following two heuristic approaches; see Figures 8 and 9 for the flowcharts. However, we cannot publish the pseudocodes since this is part of an industrial research. Our algorithms were implemented as an experimental extension to the APlus software, which is a commercial software.

Heuristic 1. This approach reduces the number of feasible tool vector considered for each tool-path point. Instead of considering a number of tool vectors for all the tool-path points, we consider multiple feasible tool vectors for only the tool-path points where the deposition head collides with the object. See Figure 4.

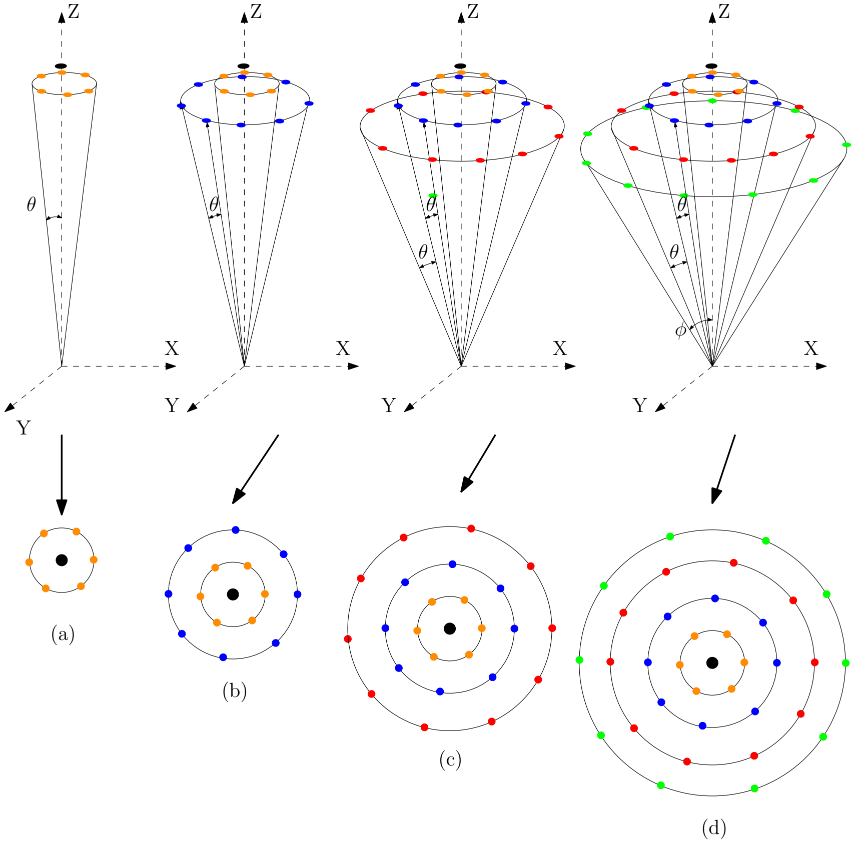


Figure 4: Illustration for heuristic approach 1.

In our previous algorithm [12], we generate feasible tool vectors for the colliding tool-path points as shown in Figure 2. Now, in this heuristic approach, we first consider feasible tool vectors for a colliding tool-path point that are closest to the original tool vector (the orange points in Figure 4) and create the configuration graph. If a collision-free tool-path is generated from the configuration graph, we return it. Otherwise, we consider more feasible tool vectors for the collision points that are farther from the original tool vector (i.e., the blue, red, and green points, respectively, in Figure 4). Instead of creating the configuration from the start for each such step, we add the new vertices and respective edges to the already created configuration graph. We then run the algorithm to find a path from s to t again.

In this way, we keep on adding more feasible tool vectors as vertices of the configuration graph, until we find a collision-free tool-path or no more feasible tool vectors remain for the collision points. In the later case, if no collision-free path is found at that point, the algorithm declares that it could not find a collision-free path.

Heuristic 2. This approach reduces the size of the configuration graph by reducing the number of tool-path points considered. Instead of including the entire tool-path, we sample tool-path points at regular interval (e.g., every tenth or every fifth point) for the graph, the difference between tool vectors angles are adjusted accordingly. Figure 5 shows an example where every fifth point is considered on the tool-path.

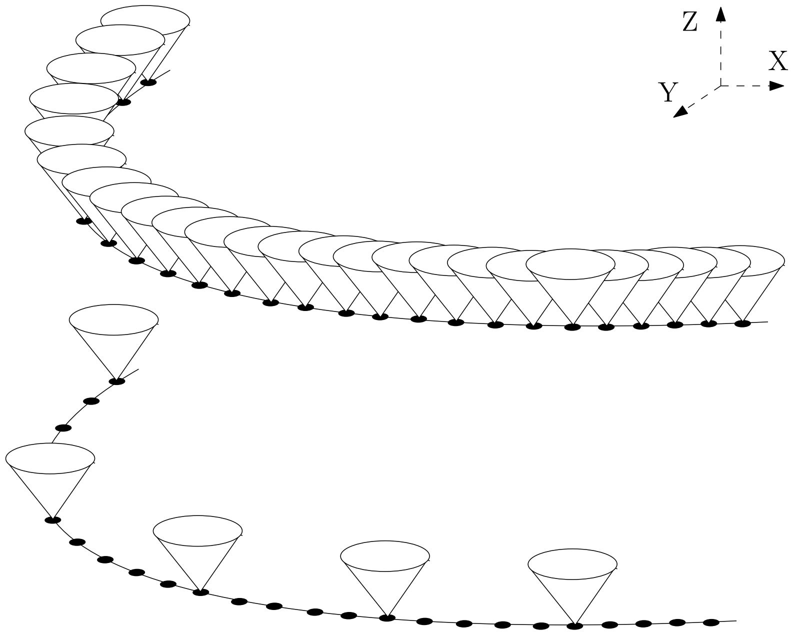


Figure 5: Illustration for heuristic approach 2: the algorithm in [12] generates feasible tool vectors (inside the conic area) for all the tool-path points (top), whereas this heuristic approach considers the feasible tool vectors for tool-path points in regular intervals (every fifth point in the bottom figure).

If a collision-free tool-path is found, we then move into each interval and find the collision-free tool-path for each interval. If the tool-path in all intervals abide by the mechanical specification of the AM machine, we report the constructed path as the final result.

We have to consider carefully how to choose the interval in the tool-path points. Ideally, we would prefer small intervals to get more accurate results. However, smaller interval does not reduce the time required by the heuristic algorithm by much. Therefore, we need to find the balance between calculating a collision-free tool-path and the time required to do that, which would definitely depend on the input tool-path given to the algorithm. Our proposed heuristic approach is certainly adaptive to that scenario as it can be implemented to find that balanced interval by running it for different intervals of tool-path points.

Note that the heuristic approaches can be applied separately, or together to improve the running time.

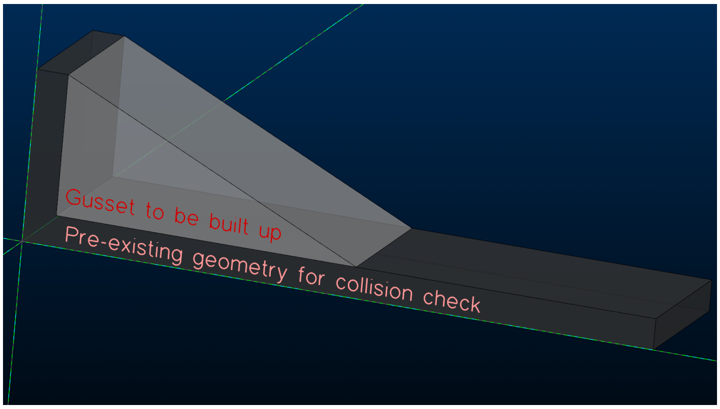
5 EXPERIMENTAL RESULTS

We applied both the above heuristics to the algorithm [12], mentioned simply as "Heuristic method" in the table below, and tested the performance on two models shown in Figure 6. The software used to visualize and generate the tool-paths for the Additive Manufacturing (AM) purposes was Mastercam and APlus, both of which are industrial standard commercial softwares. APlus software was used to connect to Mastercam. System specification is as follows:

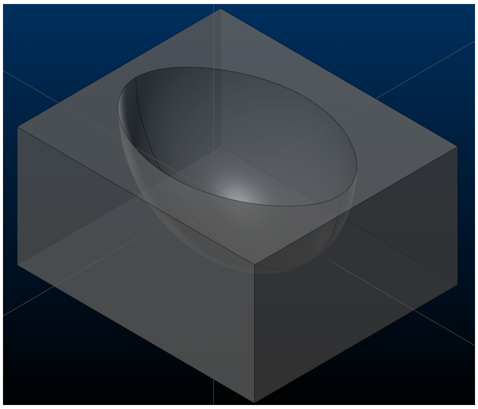
- CPU: Intel(R) Core(TM) i9-10885H CPU @2.40GHz
- Memory: 32GB 2933MHz DDR4
- Storage: 953.86GB (Model: SKHynix_HFS001TD9TNI-L2B0B)

Models	Points on the tool-path	Möller's technique only	Heuristic method
Bathtub	273 points	428 s	290 s
L-bracket	566 points	148 s	106 s

Table 1: Comparing the time to calculate the collision-free tool-path using Möller's technique for calculating the collision, with and without the heuristic method.



(a)



(b)

Figure 6: (a) The Bathtub model, and (b) the L-bracket model.

We used the Möller's to determine the collision. In this Möller's approach, the triangles of the mesh of each object are considered and the collision is determined if the triangles of the meshes are colliding. Table 1 shows the time for generating the collision-free tool-path with and without the heuristic method. The generated tool-path is presented in Figure 7.

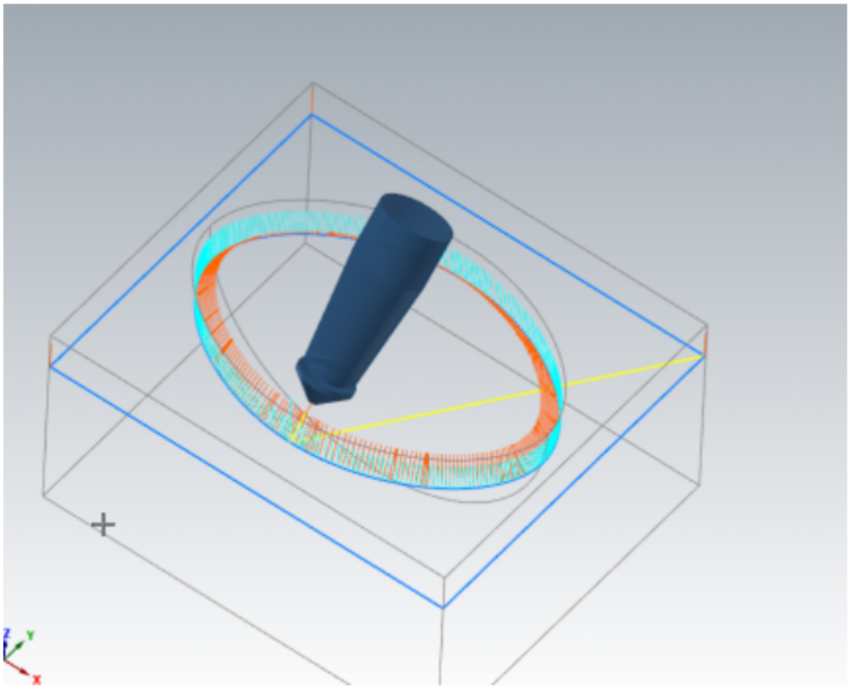


Figure 7: The collision-free tool-path generated for the Bathtub model by the heuristic methods.

6 LIMITATION

Our algorithm in this paper is heuristic, and as such, it will not be deterministic. This means that there will be cases that there might exist collision-free tool-paths, but this technique reports there is none. In other words, this approach can create false negatives.

This approach may take the same amount of time as the original algorithm in [12] based on the number of times each of heuristics 1 and 2 may need to be rerun.

7 FUTURE WORK

Currently, the main bottleneck of our approach is the Möller's technique which calculated the collision between the mesh of two given objects (the object being printed and the holder). As such, the next step is to calculate the collision between two components with a more efficient approach.

In our experiments, we exported a coarse approximation of the holder model from Mastercam and then computed its convex hull. which improved the performance of our algorithm. We believe that applying remeshing techniques, the size (i.e., the number of triangles) of the meshes could be reduced further, thus improving the running time of our algorithm.

Another technique to reduce the time for collision detection could be the *slicing and clipping* algorithm presented in [14, 16]. The idea is to compute 'slices', which are two dimensional polygons, for both the holder and the printing surface. Then for each pair of slices at a common layer or plane, one slice from the holder and one from the printing surface, we compute the intersection of the polygons to see if there is any collision.

In this paper, we considered heuristic approaches with the goal of reducing time to compute collision-free tool-paths. However, our approach may falsely decide that certain tool-path points create collision because we are using approximated models of the holders and mesh inputs for both the printed surface and the holder in collision detection. It will be an interesting direction of research to find heuristic approaches that reduce such false collision detection.

Our problem was based on real obstacles faced in the AM industry. Therefore, we plan to apply our algorithm in the industry and get feedback based on real industrial production.

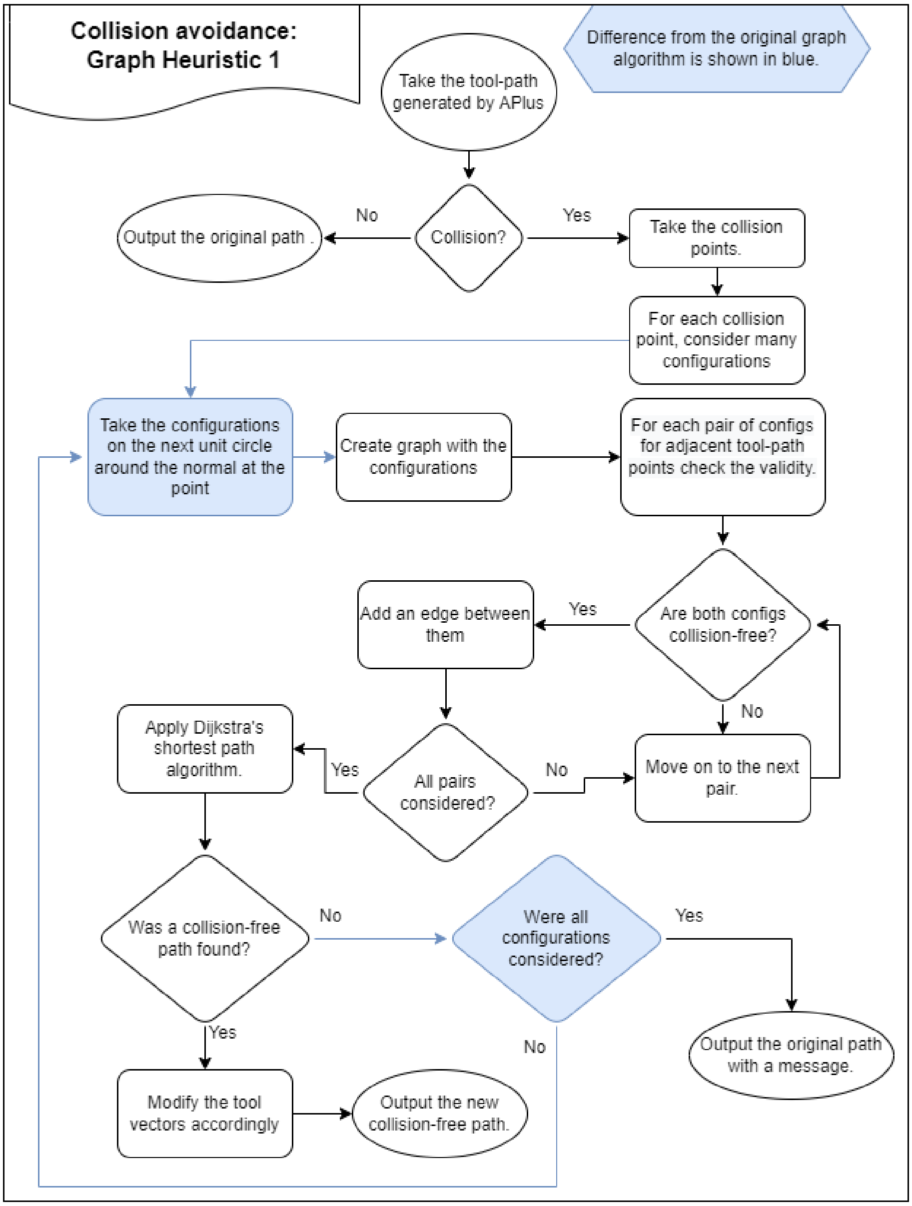


Figure 8: The flow chart of the heuristic approach 1.

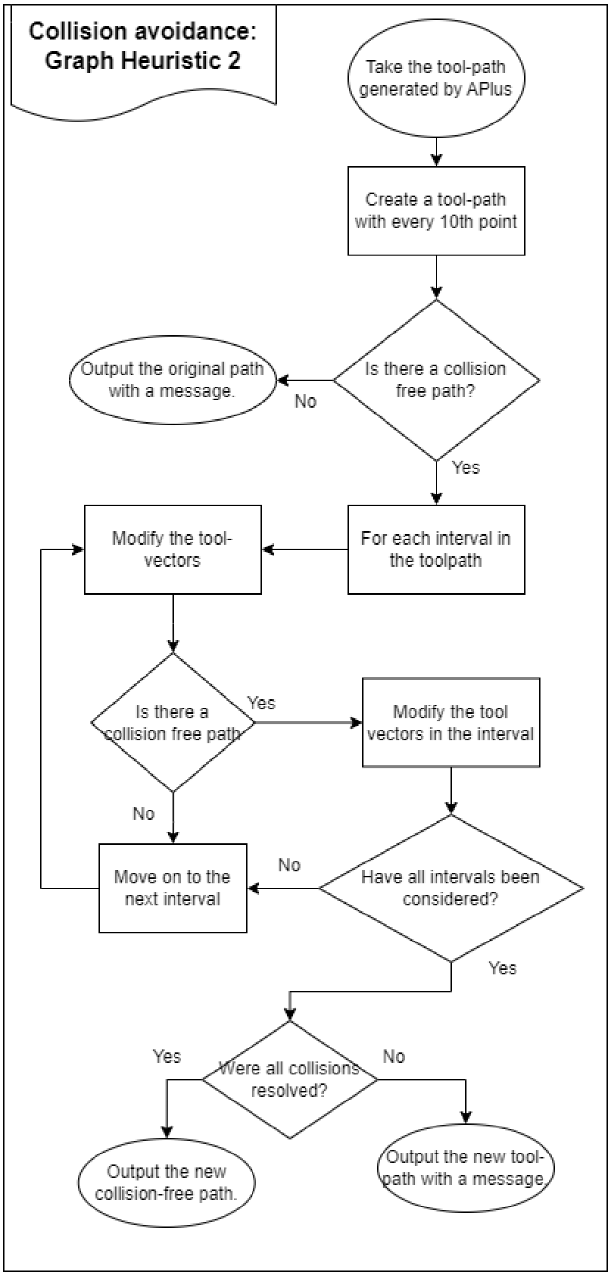


Figure 9: The flow chart of the heuristic approach 2.

8 CONCLUSION

In this work, we improved the efficiency of the fabrication of the collision-free tool-path by using heuristic methods. This was achieved by focusing on the points of the path in which collision occurs, as well as reducing the overall size of the configuration graph. Additionally, using the heuristic approach A* instead of Dijkstra's algorithm also helps efficiency.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their suggestions that helped us improve the presentation of the paper. This research is part of a project funded by Natural Sciences and Engineering Research Council of Canada (NSERC) and Ontario Centre of Innovation (OCI).

REFERENCES

[1] Aliyu, M.D.S.; Al-Sultan, K.S.: Lp-based algorithms for detecting the collision of moving objects. The Journal of the Operational Research Society, 46(7), 854–866, 1995.

[2] Balasubramaniam, M.; Laxmiprasad, P.; Sarma, S.; Shaikh, Z.: Generating 5-axis nc roughing paths directly from a tessellated representation. Computer-Aided Design, 32(4), 261–277, 2000. http://doi.org/10.1016/S0010-4485(99)00103-7.

[3] Balasubramaniam, M.; Sarma, S.E.; Marciniak, K.: Collision-free finishing toolpaths from visibility data. Computer-Aided Design, 35(4), 359–374, 2003. http://doi.org/10.1016/S0010-4485(02)00057-X.

[4] Devillers, O.; Guigue, P.: Faster Triangle-Triangle Intersection Tests. Tech. Rep. RR-4488, INRIA, 2002.

[5] Fang, G.; Zhang, T.; Zhong, S.; Chen, X.; Zhong, Z.; Wang, C.C.L.: Reinforced fdm: Multi-axis filament alignment with controlled anisotropic strength. 39(6), 2020. http://doi.org/10.1145/3414685.3417834.

[6] Held, M.: Erit - a collection of efficient and reliable intersection tests. Journal of Graphics Tools, 2, 25–44, 1998. http://doi.org/10.1080/10867651.1997.10487482.

[7] Jiang, Z.; Wang, H.; Sun, Y.: Improved co-scheduling of multi-layer printing path scanning for collaborative additive manufacturing. IISE Transactions, 53(9), 960–973, 2021. http://doi.org/10.1080/24725854.2020.180.

[8] Li, L.L.; Zhang, Y.F.: Flat-end cutter accessibility determination in 5-axis milling of sculptured surfaces. Computer-Aided Design and Applications, 2(1-4), 203–212, 2005. http://doi.org/10.1080/16864360.2005.10738368.

[9] Li, L.L.; Zhang, Y.F.: An integrated approach towards process planning for 5-axis milling of sculptured surfaces based on cutter accessibility map. Computer-Aided Design and Applications, 3(1-4), 249–258, 2006. http://doi.org/10.1080/16864360.2006.10738462.

[10] Möller, T.: A fast triangle-triangle intersection test. Journal of Graphics Tools, 2, 25–30, 1997. http://doi.org/10.1080/10867651.1997.10487472.

[11] Nishat, R.I.; Bahoo, Y.; Georgiou, K.; Hedrick, R.; Urbanic, R.J.: Collision-free multi-axis tool-path for additive manufacturing. In Proceedings of CAD'22, Beijing, China, July 11-13, 297–301, 2022. http://doi.org/10.14733/cadconfP.2022.297-301.

[12] Nishat, R.I.; Bahoo, Y.; Georgiou, K.; Hedrick, R.; Urbanic, R.J.: Collision-free multi-axis tool-path for additive manufacturing. Computer-Aided Design and Applications, 20(6), 1094–1109, 2023. http://doi.org/10.14733/cadaps.2023.1094-1109.

[13] Plakhotnik, D.; Glasmacher, L.; Vaneker, T.; Smetanin, Y.; Stautner, M.; Murtezaoglu, Y.; van Houten, F.: Cam planning for multi-axis laser additive manufacturing considering collisions. CIRP Annals, 68(1), 447–450, 2019. http://doi.org/10.1016/j.cirp.2019.04.007.

[14] Tang, T.; Bohez, E.L.; Koomsap, P.: The sweep plane algorithm for global collision detection with workpiece geometry update for five-axis nc machining. Computer-Aided Design, 39(11), 1012–1024, 2007. http://doi.org/10.1016/j.cad.2007.06.004.

[15] Tang, T.D.: Algorithms for collision detection and avoidance for five-axis nc machining: a state of the art review. Computer-Aided Design, 51, 1–17, 2014. http://doi.org/10.1016/j.cad.2014.02.001.

[16] Tang, T.D.; Bohez, E.L.J.: A new collision avoidance strategy and its integration with collision detection for five-axis nc machining. The International Journal of Advanced Manufacturing Technology, 81, 1247–1258, 2015.

[17] Wang, N.; Tang, K.: Automatic generation of gouge-free and angular-velocity-compliant five-axis toolpath. Computer-Aided Design, 39(10), 841–852, 2007. http://doi.org/10.1016/j.cad.2007.04.003.

[18] Xu, X.J.; Bradley, C.; Zhang, Y.F.; Loh, H.T.; Wong, Y.S.: Tool-path generation for five-axis machining of free-form surfaces based on accessibility analysis. International Journal of Production Research, 40(14), 3253–3274, 2002. http://doi.org/10.1080/00207540210150643.