Solid Freeform Fabrication 2024: Proceedings of the 35th Annual International
Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference
Reviewed Paper

Reducing Travel Moves in Material Extrusion Additive Manufacturing Through Graph Theory

Logan J. Hutton*, Joseph Bartolai*

*Applied Research Laboratory
The Pennsylvania State University, University Park, PA

Abstract

A novel graph-theory driven toolpath generation algorithm for polymer Material Extrusion (MEX) additive manufacturing (AM) that considers process optimization when writing toolpaths is presented. Contemporary toolpath design algorithms are written to prioritize robustness (always reaching a valid solution) and computational cost (time to solution). This novel toolpath generation algorithm, known as GRATER (GRAph Theory based slicER), is a work-in-progress tool that uses graph theory to guarantee a single continuous toolpath (a toolpath with no travel moves) for every closed contour. GRATER reduced travel moves, which create defects within MEX parts particularly those built using pellet-fed MEX, by up to 95% when compared to contemporary slicing softwares. Additionally, GRATER is shown to reduce travel distance by a factor of 3 and build time by 25%.

Motivation

Big Area Additive Manufacturing (BAAM) has a variety of other names such as Large Area Additive Manufacturing (LAAM) or Wide Area Additive Manufacturing (WAAM) — WAAM has fallen out of fashion due to confusion with Wire Arc Additive Manufacturing that shares the same acronym. BAAM machines have "big" build volumes and use pellets for the material feedstock as a cost-effective alternative to filament, see Figure 1 [1]. Filament MEX systems have a positive displacement extrusion system where the volume of filament pushed into the hotend is assumed to be directly correlated to the volume of molten plastic extruded. By assuming the cross-section of the extrudate is a stadium, the slicer can accurately model the dimensions of the extruded road. [2, 3]. Pellet MEX systems are, however, not positive displacement with respect to the volume of plastic extruded. This small difference has created a litany of issues including the development of controller strategies that attempt to correlate changes in screw RPM to changes in the volume of plastic extruding [4]. In practice, this limits pellet systems to empirically determining parameters such as bead width, screw RPM, and layer height for a steady state flow rate regime [3]. The print quality degrades when not at a steady state, this can include decelerating for sharp corners or extrusion starts and stops [4]. The inability for precise extrusion control means that pellet systems cannot retract to stop extruding molten plastic and travel as filament systems are able. This small difference leads to challenges in material deposition consistency, mechanical properties, geometric accuracy, and in almost every aspect of process planning. Compounding this effect is that conventional toolpath algorithms do not attempt to minimize the travel moves, which are known to lead to defect-riddled parts when used in conjunction with pellet systems. These defects are known as travel defects. The tools developed in this paper, seek to reduce travel moves and therefore travel defects.

Figure 1: World's largest 3D printer at UMaine's Advanced Structures and Composites Center with a build volume of 100×22×10 feet, reproduced from [5].

MEX AM Toolpath Design

There have been various attempts to create an algorithm that produces the optimal toolpath, with optimal being defined as the toolpath that gives the highest strength for a specific load case or the toolpath that completely fills space. Xia et al. (2020) designed a stress-based toolpath algorithm that constructs toolpaths parallel with the maximum principal stress directions with a depth-first search and then used Dijkstra's algorithm to minimize travel distance [6]. The limitation of Xia et al.'s (2020) work is the algorithm only considers 2D complex shapes with a single load case [6]. Shaikh et al. (2016) developed a toolpath algorithm that follows the Hilbert Curve, which is a continuous space-filling fractal [7]. This approach is implemented in most slicers and can be selected as the Hilbert Curve infill pattern. Yoo et al. (2020) used a Monte Carlo Tree Search algorithm to efficiently fill space and reduce travel moves, but the method was significantly slower than conventional space-filling toolpath algorithms [8]. Gupta et al (2020) developed a framework to create a a graph that is Eulerian and hence has an Eulerian tour that can be used to find a continuous infill [9]. Most recently Borish et al. (2023) developed a toolpath algorithm that generates a single path for closed contours by using graph theory and topological hierarchy [10]. This paper takes a holistic approach to toolpath design, aspiring to create toolpath algorithms that reduce defects in parts and increase mechanical part properties.

The work presented in this paper differs from previous work in two main ways. At the heart of GRATER is a method of taking 2D meshes and transforming them into graphs and finding the longest path in that graph. The graph transformation importantly does not seek to find a Eulerian mapping like the one from Gupta et al (2020) [9]. Eulerian tours are fundamentally ill suited for large scale MEX AM as they visit nodes more then once leading to a collision between the nozzle and the previously deposited material. This collision is trivial on the small scale, but on large format machines this can and does cause build failures. Hence the first way that GRATER is different is that the algorithms in GRATER were built from the ground up for large format MEX AM. Secondly GRATER is a slicer and as a whole seeks to reduce travel moves and travel distance. This includes things like a greedy algorithms for travel moves when multiple closed contours are in a single layer and reordering toolpaths to reduce travel moves between layers. For a proper deep dive into the algorithms in GRATER see the author's thesis [11].

ORNL `Slicer 2` is an evolution of ORNL `Slicer`, which was the first purpose-built slicing software for large-scale printing [12]. `Slicer 2`'s current approach for travel moves is to try and minimize their effect on print quality by modifying the beginning and ending of travel moves. For instance, a *Tip Wipe*, which is a motion used to wipe the tip of the nozzle and break away the extrusion bead, is implemented at the beginning of travel moves [13]. This addressing of the symptoms caused by travel moves is effective, enough to consistently get adequate parts, up to a point. The tools presented in this paper seek to address the root of the problem, but future work that combines both approaches could greatly improve the chronic inconsistency of pellet MEX AM parts.

Graph Theory

Graph theory is a relatively new mathematical field that models the pairwise relation between objects. A graph is defined by an ordered pair (V,E) where V is a finite, nonempty set called *vertices*, and E is a set of unordered pairs of vertices called *edges* [14]. A vertex u is adjacent to a vertex v if the unordered pair $\{u,v\}$ is in E, i.e., the vertices are *connected* to one another via the edge $e \in E$ [15]. Graphs are commonly presented visually by representing each node as a point in a plane and each edge as a line connecting the vertices [16]. An example of this visual representation can be seen in Figure 2, where each node is labeled with a number, and the edges connecting the nodes are denoted as solid lines. An example of the set notation from Figure 2 would be the edge between nodes 2 and 7 being denoted in the edge set E as $\{2,7\}$ or as $\{7,2\}$ since edges are unordered pairs. The *degree* of a vertex is defined as the number of incident edges (edges that connect to the node) [17]. For example, every node in the Peterson graph, depicted in Figure 2, has a degree of 3.

The geometric position of the points, the path the edges take between nodes, and the length of the edges (typically) hold no meaning. The graph is usually drawn such that no edges intersect other edges, to aid in human readability [20]. A graph that can be drawn without such intersection is defined as a planar embedding of that graph and said graph is considered a *planar graph* [17]. A geometric graph is a graph whose nodes are defined by geometric means [21]. For our purposes, we will be discussing geometric graphs in the Euclidean plane, but other topological geometric graphs have a deep depth of research and practical applications [21]. Nodes in this work will be defined by their coordinates in a 2D coordinate space and the edges will be the unordered pair of the indices of the nodes [19]. The *edge list* will be referred to as the *adjacency list*; an example of an adjacency list and its graphical representation is shown in Figure 3.

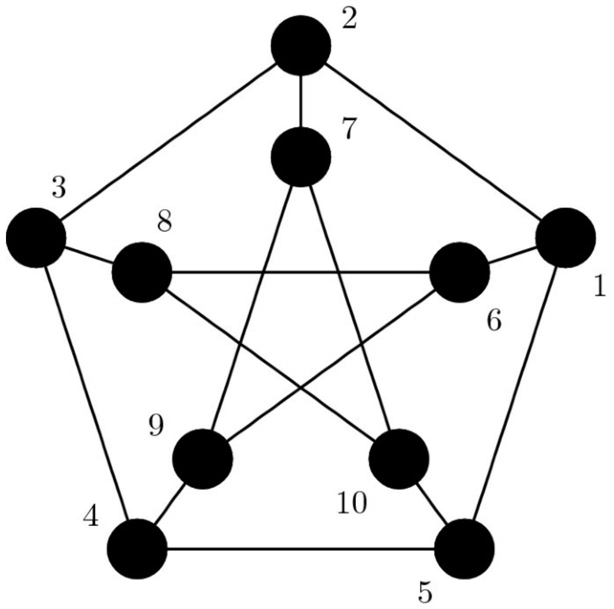
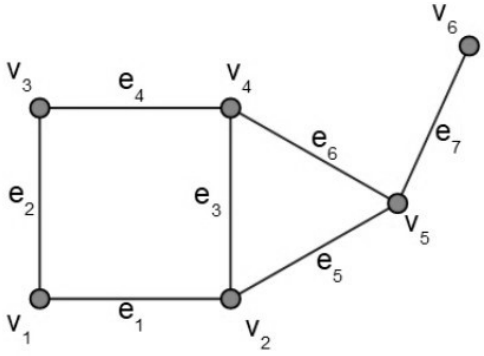


Figure 2: Peterson Graph with nodes labeled 1-10, reproduced from [18].

	v_1	v_2
e_1	1	2
e_2	1	3
e_3	2	4
e_4	3	4
e_5	2	5
e_6	4	5
e_7	5	7

(a) Adjacency List



(b) Graphical Representation

Figure 3: An adjacency list and its corresponding graphical representation. Each node is defined in a separate list as a vertex in 2D space. Reproduced from [19].

For practical applications, it is often useful to assign a weight to the edges to represent some cost or price, commonly referred to as distance in geometric graphs, to traverse that edge. Weights are used in a variety of applications such as connecting landline nodes or planning roads as they directly map to physical distances in the system [14]. A common practical problem to solve with a weighted graph is to find the shortest distance between two vertices within a graph. As such, a variety of robust algorithms have been developed to solve the so-called shortest path problem [14]. A popular greedy algorithm, an algorithm that makes the locally optimal choices at each decision step to solve the problem, to find the shortest distance is Dijkstra's algorithm. [15]. This heuristic strategy rarely results in the optimal solution but it does, in many cases, provide a near-optimal solution significantly faster than the optimal solution could be found [15]. Dijkstra's algorithm is popular in solving the shortest path problem due to its runtime of $\mathscr{O}(|V^2|)$, or in other words, the runtime of the algorithm scales by the number of vertices in a graph squared [15]. There are faster shortest-path algorithms for certain types of graphs, but Dijkstra's is considered the fastest shortest-path algorithm that can handle any graph well. We will use these robust shortest-path algorithms as a framework for the toolpath algorithm that will allow the algorithm to robustly handle complex layer geometry.

GRATER

THIS work has developed a suite of tools collectively known as the **GRA**ph **T**heory based slice**R** (GRATER) that seeks to improve the MEX AM process by reducing travel moves. As discussed in the previous section, travel defects are inherent to travel moves; therefore, reducing travel moves will reduce travel defects, reduce the amount of post-processing necessary, and reduce print time. The reduction of travel moves is especially relevant to the pellet extrusion systems found on large format MEX AM systems due to their inherent difficulties in extrusion control as discussed in the previous section. This chapter will discuss the structure and algorithms used by GRATER in detail with the next chapter will discuss the results of using GRATER to manufacture parts.

GRATER was written in `MATLAB` and currently ingests ASCII STLs, binary STLs, or NaN-separated contours (similar to some laser additive manufacturing processes). GRATER currently outputs `Marlin` flavored Gcode for filament MEX AM systems or for the Juggerbot 3D's P3-44. The P3-44 uses a pellet-fed screw extrusion system that can output up to fifteen pounds of filament an hour [22]. The P3-44 runs a limited version of `Marlin`, with the main difference being that a screw RPM is commanded instead of an extrusion value. The structure of GRATER is described in Figure 4. Each black box in Figure 4 represents a function in `MATLAB` and the inputs and outputs represent the variables being passed into and out of the functions. The blue boxes are the rest of the MEX AM process that are either an input into GRATER or the output of GRATER. Each function was discussed at length in the first Authors thesis, this paper serves as a quick look into what GRATER can do without getting bogged down in the minute details [11].

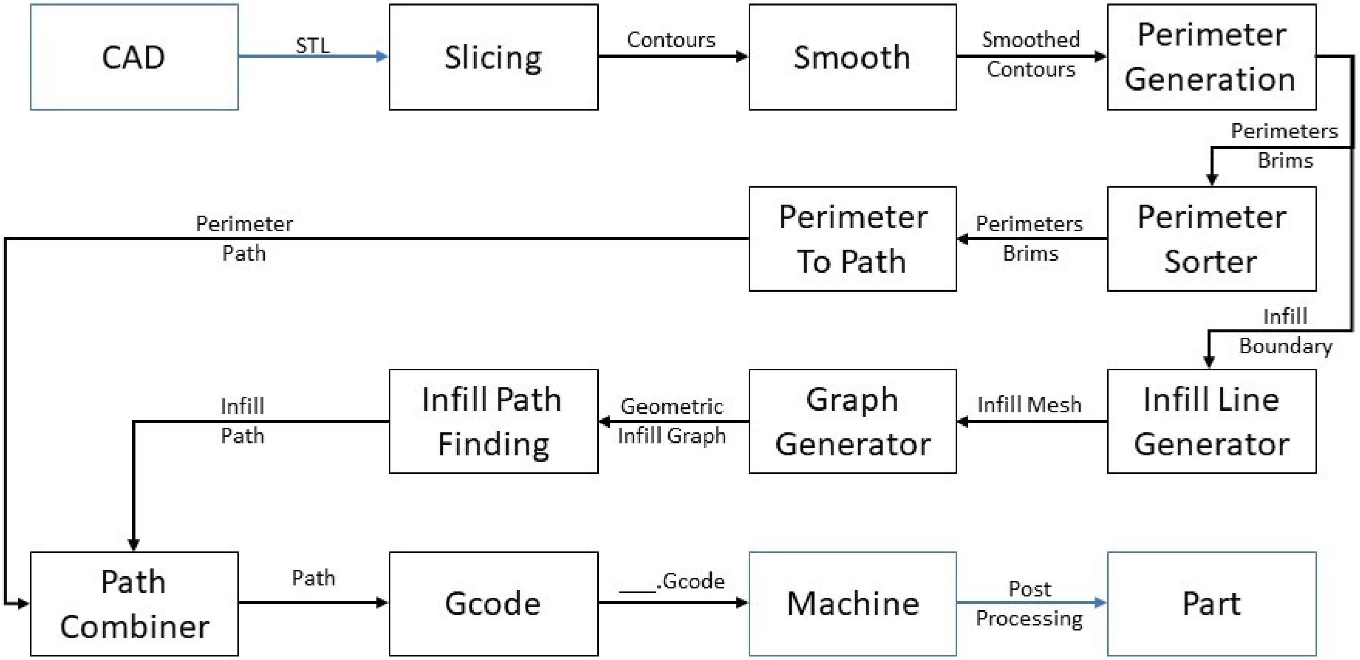


Figure 4: This diagram represents the structure of GRATER. Every black box is a separate `MATLAB` function and the inputs and outputs of those boxes is the major data being passed between functions. The blue boxes represent the surrounding steps in the MEX AM process with the blue inputs and outputs being inputs and outputs to GRATER.

(a) `Slic3r` (b) GRATER

Figure 5: A 2.7 inch gyroid section sliced using `Slic3r` with retraction off versus one processed with GRATER on a desktop filament MEX AM system. Note the excessive amount of stringing between the two regions as a result of excessive travel moves.

Results & Discussion

To determine if GRATER achieves its goal of reducing travel defects, we will first test GRATER on a small scale filament extrusion MEX AM machine. Then move on to large scale testing where we will compare GRATER's travel moves, travel distance, and build time to other slicers.

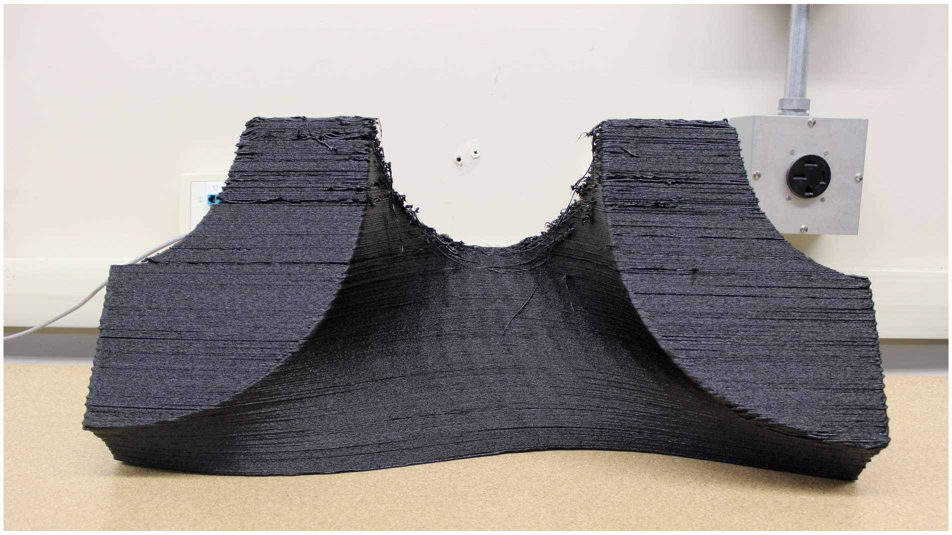
Desktop Benchmark Testing

GRATER was first tested on a desktop filament MEX AM system before moving on to large-scale pellet MEX AM system tests. A benchmark part was designed as it represents an end-use part that is extremely difficult to print on large-scale MEX AM systems with current slicers. Retractions, when filament MEX AM systems retract filament from the hotend to prevent stringing and defects during travel moves, were disabled for both prints to simulate the inability of a large-scale pellet MEX AM system to stop extruding plastic during travel moves. As seen in Figure 5(a), the `Slic3r` sliced part has significant stringing between the two peaks at the top of the print. The `Slic3r` sliced part also has stringing on the bottom layers similar to Figure 6(a), but this stringing is difficult to capture in an image. The GRATER sliced part in Figure 5(b) has slight stringing between the two peaks at the top of the print since it has to travel across the empty space once per layer; this stringing was insignificant compared to the `Slic3r` part. The bottom of the parts were similar. The GRATER sliced part had no stringing on the bottom layers, unlike the `Slic3r` part.

Large-Scale Benchmark Performance

After the success of the small-scale prints, seen in Figure 5, the same shape was made at full scale on a Juggerbot P3-44 large-scale pellet-fed MEX AM system. `Simplify3D`, a slicer similar to `Ultimaker Cura` and `Slic3r`, was used instead of `Slic3r` because `Simplify3D` is the manufacturer recommended slicer with a `Simplify3D` profile being provided by Juggerbot. The `Simplify3D` part failed due to the print head crashing into the print while traveling, but the point of comparison for travel defects is evident with the unfinished benchmark part. This failure is consistent with previous attempts to print this and similar geometries on the P3-44 with the `Simplify3D` slicer. The excessive travel moves used by `Simplify3D` and other contemporary slicers lead to build failures due to excessive stringing and travel defects. The part was printed

(a) `Simplify3D`



(b) GRATER

Figure 6: A 27 inch gyroid section sliced with `Simplify3D` and GRATER on a pellet MEX AM system. The `Simplify3D` build failed due to travel defects from excessive travel moves.

with no infill, as no top surface needed infill to support it. GRATER would perform even better if infill was included as we will see later. The GRATER part was completed with minimal stringing between the two peaks and no stringing along the bottom of the part that consists of *a single* contour. Figure 6 showcases this dramatic improvement.

GRATER guarantees one continuous path per layer and starts the next layer near where the previous layer ended so that the single contour layers only have one small travel move. This is in stark contrast to the multiple travel moves per layer that the `Simplify3D` sliced part had, which led to the abundance of stringing seen in Figure 6(a). After these prints, the Juggerbot P3-44 underwent a series of firmware and hardware upgrades that came along with a profile for Oak Ridge National Lab's `Slicer` [12, 13]. Therefore the following tests were done with `ORNL Slicer` instead of `Simplify3D`.

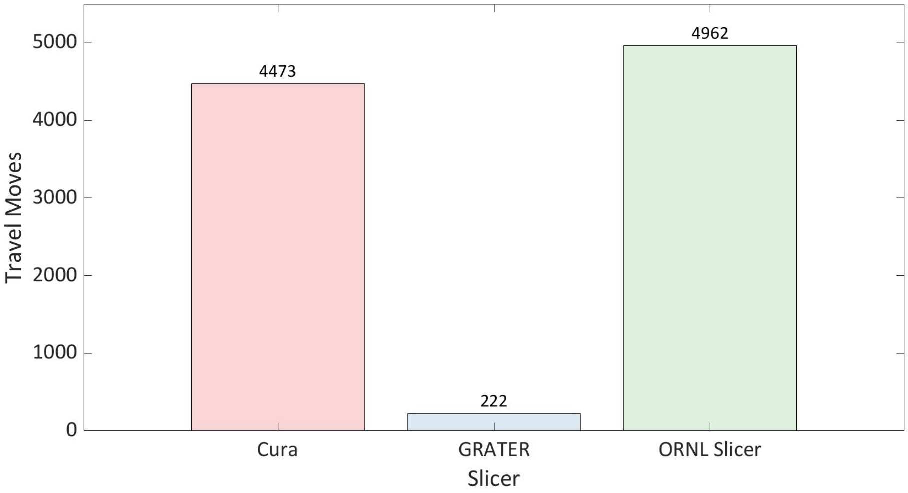


Figure 7: Total travel moves of the lattice section on the Juggerbot P3-44. GRATER has an order of magnitude fewer travel moves than both `Cura` and `Slicer`.

GRATER was built with the aspiration of reducing travel defects by reducing both the number of travel moves and the length of travel moves. Every travel move invalidates the steady-state extrusion flow condition that the process parameters were empirically determined under. This leads to defects such as over and under-extrusion before and after every travel move. Molten plastic continually oozes out of the nozzle during travel moves in pellet-fed MEX AM systems. This ooze can even cause prints to fail such as in Figure 6(a). To test if GRATER lives up to aspirations, it will be compared against the manufacturer-provided slicer (`ORNL Slicer`) and the most popular open-source slicer (`Ultimaker Cura`).

A new end-use part was used for the following tests and can be seen in Figure 8. The same geometry was sliced in three slicers (`Cura`, GRATER, and `ORNL Slicer`) for the Juggerbot P3-44. Slice parameters were held the same across all of the slicers to isolate the effect of toolpath planning. The part was 221 layers tall and had a single contour on every layer. GRATER had one travel move per layer plus one travel move to move from home to the start of the first layer. Figure 7 shows the number of travel moves each slicer had for the same part. GRATER has *an order of magnitude fewer* travel moves than both `Ultimaker Cura` and `ORNL Slicer` for the geometry tested.

This reduction in travel moves leads to a reduction in build time, specifically on the Juggerbot P3-44. The P3-44 pauses every single time the extruder is stopped or started, which happens during every travel move. As is evident from Figure 9, these pauses result in *thousands* of stop-start events that aggregate into more than a 20% increase in total build time for both `Cura` and `Slicer` as compared to GRATER's continuous toolpath planning algorithm for the geometry tested. The `Cura` build time is, however, an estimate based on the expected build time as no benchmark parts were able to be successfully printed. The estimate is based on the time it took to print the failed part, combined with the estimated build time. This experiment clearly shows the benefit of reducing start-stop events on these pellet MEX AM systems to increase part production rates.

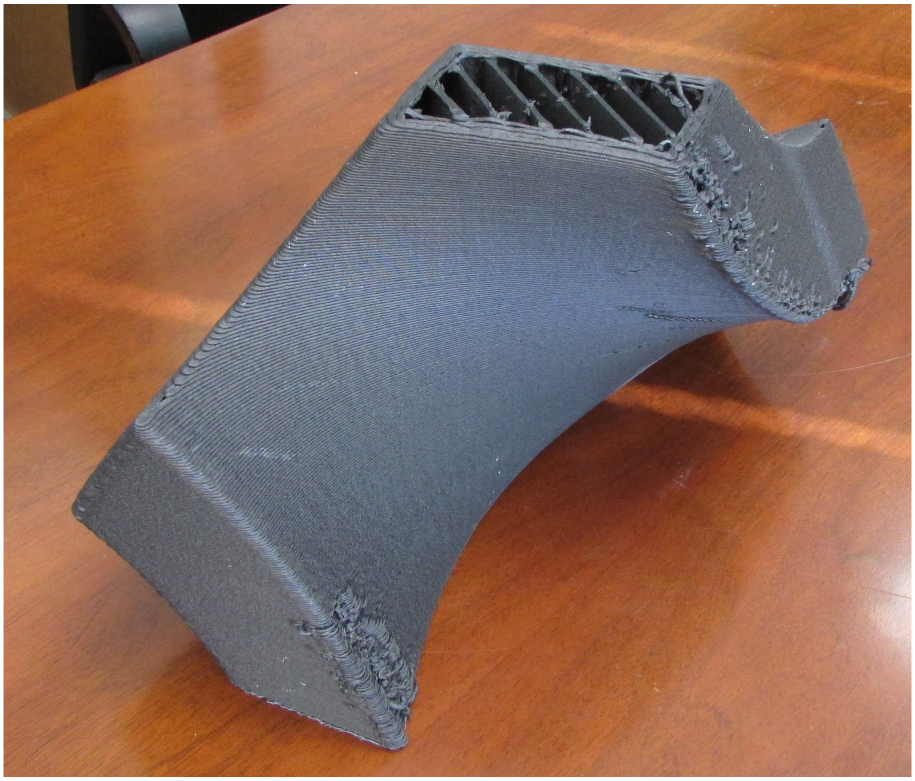


Figure 8: This test part was chosen as it was an end-use part that had a variety of cross-sections through the build direction.

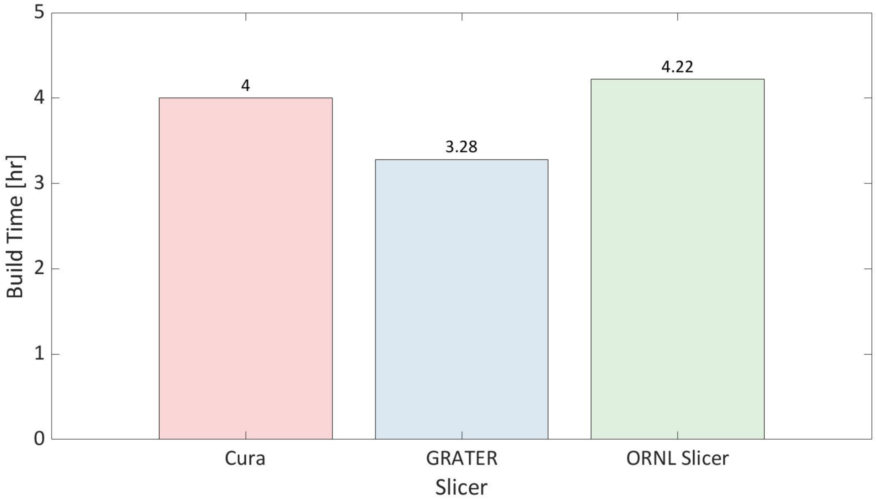


Figure 9: Build times of the lattice section on the Juggerbot P3-44. The Cura build time is an estimation as no Cura sliced part was printed successfully. The reduction in travel moves has the benefit of reducing print time on the Juggerbot P3-44.

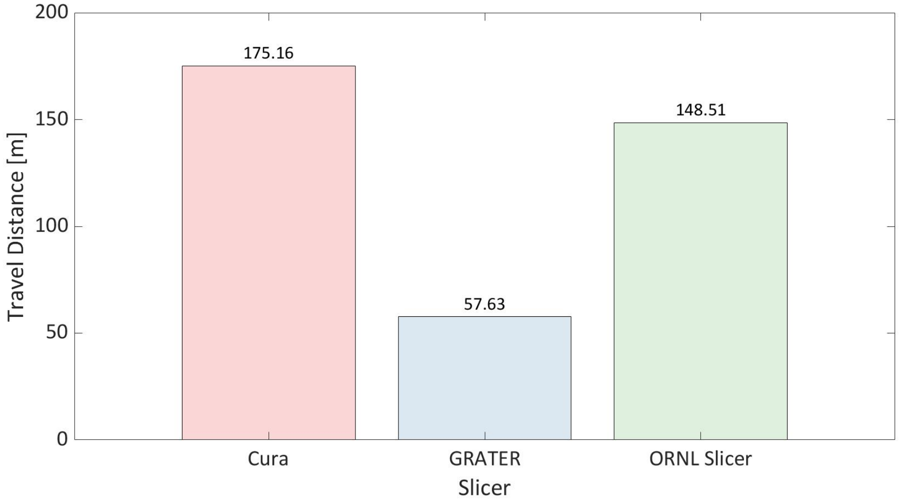


Figure 10: Travel distance of the lattice section on the Juggerbot P3-44. GRATER reduces the total distance traveled, reducing the amount of stringing due to traveling.

Pellet MEX AM systems ooze material as they travel leading to defects, GRATER sought to try and reduce the distance traveled to mitigate this problem. To calculate travel distance a `MATLAB` script was developed for each of the three slicers, which can be seen in the first authors' thesis [11]. Each of the three slicers denotes travel moves differently, making a script for each one necessary. The distance was the linear distance in the (x, y) from the starting location to the location being traveled. All three slicers implement a z-hop, an upwards travel in z to avoid crashing into the part when traveling, but this distance was not calculated due to it not contributing to the ooze defect. GRATER reduced the total distance traveled by a factor of three for the geometry tested.

Summary

GRATER was built to reduce travel defects, specifically to make the pellet MEX AM process more reliable by reducing the travel defects inherent to the process. GRATER accomplished this by utilizing graph theory to find continuous toolpaths. GRATER was successful in this aspect as measured against both a contemporary filament MEX AM slicer and the manufacturer's recommended slicer in the following three metrics for the geometry tested.

1. GRATER reduced travel moves by an order of magnitude or 95%.
2. GRATER reduced total distance traveled by a factor of three.
3. GRATER reduced the print time by approximately 25%.

Future work will look to build out GRATER into a fully featured slicer or implement the graph theory path finding algorithm in a fully featured slicer.

References

[1] “Review of current problems and developments in large area additive manufacturing (LAAM),” pp. 1539–1548. [Online]. Available: https://repositories.lib.utexas.edu/bitstream/handle/2152/90746/2021-127-Crisp.pdf?sequence=2

[2] Slic3r manual – welcome to the slic3r manual. [Online]. Available: https://manual.slic3r.org/

[3] A. Roschli, M. Borish, L. White, C. Adkins, C. Atkins, A. Barnes, B. Post, Z. DiVencenzo, C. Dwyer, G. Rudiak, and B. Zellers, “Transmitting g-code with geometry commands for extrusion additive manufacturing,” in *Proceedings of the 34th Annual International Solid Freeform Fabrication Symposium - An Additive Manufacturing Conference*, vol. 34, pp. 950–958.

[4] P. Chesser, B. Post, A. Roschli, C. Carnal, R. Lind, M. Borish, and L. Love, “Extrusion control for high quality printing on big area additive manufacturing (BAAM) systems,” vol. 28, pp. 445–455. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214860418307000

[5] edalton. UMaine 3d prints two new large boats for u.s. marines, breaking previous world record. [Online]. Available: https://composites.umaine.edu/2022/04/11/umaine-3d-prints-two-new-large-boats-for-u-s-marines-breaking-previous-world-record/

[6] L. Xia, S. Lin, and G. Ma, “Stress-based tool-path planning methodology for fused filament fabrication,” vol. 32, p. 101020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214860419315659

[7] S. Shaikh, N. Kumar, P. K. Jain, and P. Tandon, “Hilbert curve based toolpath for FDM process,” in *CAD/CAM, Robotics and Factories of the Future*, D. K. Mandal and C. S. Syan, Eds. Springer India, pp. 751–759, series Title: Lecture Notes in Mechanical Engineering. [Online]. Available: https://link.springer.com/10.1007/978-81-322-2740-3_72

[8] C. Yoo, S. Lensgraf, R. Fitch, L. M. Clemon, and R. Mettu, “Toward optimal FDM toolpath planning with monte carlo tree search,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4037–4043. [Online]. Available: https://ieeexplore.ieee.org/document/9196945/

[9] P. Gupta, B. Krishnamoorthy, and G. Dreifus, “Continuous toolpath planning in a graphical framework for sparse infill additive manufacturing,” *Computer-Aided Design*, vol. 127, p. 102880, Oct. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0010448520300737

[10] M. Borish, A. Roschli, C. Wade, B. Post, L. White, and C. Adkins, “Single path generation for closed contours via graph theory and topological hierarchy,” in *34th Annual International Solid Freeform Fabrication Symposium*, pp. 909–923.

[11] L. Hutton, “Graph theory driven toolpath design for material extrusion additive manufacturing.”

[12] A. Roschli, A. Messing, M. Borish, B. K. Post, and L. J. Love, "ORNL SLICER 2: A NOVEL APPROACH FOR ADDITIVE MANUFACTURING TOOL PATH PLANNING."

[13] "Slicer_2_user_guide.pdf." [Online]. Available: https://ornlslicer.github.io/Slicer-2/doc/Slicer_2_User_Guide.pdf

[14] L. R. Foulds, *Graph Theory Applications*, ser. Universitext. Springer New York. [Online]. Available: http://link.springer.com/10.1007/978-1-4612-0933-1

[15] J. A. McHugh, *Algorithmic graph theory*. Prentice Hall.

[16] S. Mathew, J. N. Mordeson, and D. S. Malik, *Fuzzy Graph Theory*, ser. Studies in Fuzziness and Soft Computing. Springer International Publishing, vol. 363. [Online]. Available: http://link.springer.com/10.1007/978-3-319-71407-3

[17] D. B. West, *Introduction to graph theory*, 2nd ed. Prentice Hall.

[18] "Advanced studies in behaviormetrics and data science: essays in honor of akinori okada."

[19] D. Legland, "MatGeom library user manual."

[20] "Handbook of graph drawing and visualization."

[21] J. Pach, *Towards a theory of geometric graphs*, ser. Contemporary mathematics. American mathematical society, no. 342.

[22] Tradesman series™ p3-44. [Online]. Available: https://juggerbot3d.com/products/tradesman-series-p3-44/