Solid Freeform Fabrication 2022: Proceedings of the 33rd Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference
Reviewed Paper

HYBRID CURVE FITTING FOR REDUCING MOTION COMMANDS IN OBJECT CONSTRUCTION

Charles Wade* and Michael Borish*
**Manufacturing Science Division, Oak Ridge National Laboratory, Knoxville, TN 37932*

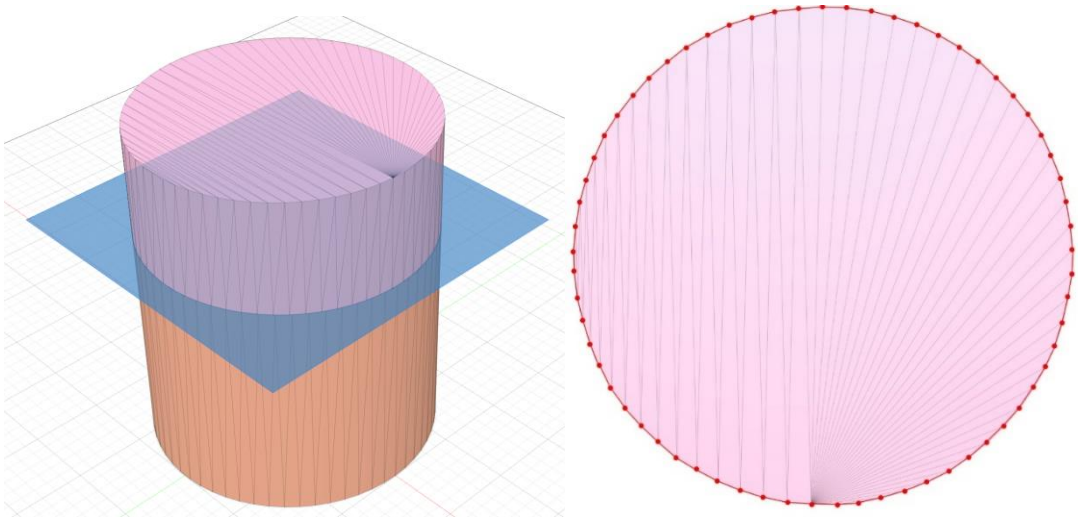
Abstract

Existing slicing software for additive manufacturing typically requires a triangulated mesh as its input. Triangulated meshes are approximate representations of exact CAD models. Despite the loss in dimensional accuracy, triangulated meshes are used because they are computationally easier to cross-section and offset than the exact geometry in CAD format. When a triangulated object is prepared, the resulting machine instructions include only linear motion commands. Numerous modern motion controllers can move in arc and spline motions; however, the absence of slicing software that supports curvature prevents these commands from being leveraged. To address this limitation, this paper presents a method for the hybrid reconstruction of arcs and splines as a post-processing step to traditional slicing. This method can greatly reduce the number of motion commands required to construct an object by printing smooth curved surfaces. This concise representation of tool-pathing allows for more even extrusion and is computed without a major impact on slicing time.

Introduction

Objects in additive manufacturing (AM) are normally constructed using sequences of line segments that define the geometry of the object. Toolpaths are commonly exported by slicing software as g-code instructions G0 and G1 that are interpreted as linear motion commands by a machine. As demonstrated in Figure 1, line segments are generated as part of the cross-sectioning process that occurs to cut 3D objects into toolpath layers [14]. Since the operations to cross-section triangulated meshes and offset polylines are simpler than the corresponding curved geometry operations, most slicing software uses an exclusively line segment-based approach. To accurately represent curved geometry, these segments use a high density of linear commands to approximate arcs and splines. The sheer number of line segments required for objects with curved surfaces results in sizeable g-code files that become infeasible for large format AM control systems. Additionally, densely packed line segments result in choppy machine kinematics that reduce both the quality and probability of a successful print. Control systems that are computationally limited may also be incapable of processing successive straight segments at high speed, increasing construction time. Furthermore, approximating curved geometry with line segments reduces the overall surface finish and tolerance of printed objects.

In this paper, we present a hybrid approach for AM toolpath planning that leverages both arc and spline commands. The goal of this work is to identify sections in AM parts that contain curvature and apply fitting methods to replace densely packed line segments with arc and spline segments. First, we examine related and foundational work. Then, we provide an overview of the individual methods used for fitting arcs and splines. Next, we present a novel hybrid approach to determine when arc versus spline fitting should be applied to a section of line segments. Finally, we compare several examples constructed with only line segments and their corresponding hybrid-fit constructions. An analysis into the impact on slicing time is also conducted to determine the feasibility of integrating this algorithm into the AM workflow.



(a) (b)

Figure 1: (a) Input triangulated mesh with cross-sectioning plane, (b) Cross-sectional layer approximating curvature with many line segments

Related Work

Traditional AM slicing uses explicitly defined triangulated surface-mesh representations for 3D geometry. Although this representation reduces computational complexity with cross-sectioning, offsetting, and filling operations, it introduces numerous inaccuracies and space limitations. The use of implicit representations in AM is the subject of numerous papers [4, 5, 18, 19]. These works present interesting solutions for slicing with implicit functions, but none present a complete set of replacement operations for traditional AM slicing. For instance, Jin et al. proposed using the implicit level-set method to smooth input contours with curvature information [6]. Although the level-set method reduced the number of empty voids when compared with explicit slicing, it was limited to certain geometries and by computational complexity. Another approach was taken by Brunton et al. using an implicit model with signed distance fields [2]. Their algorithm primarily focused on space efficient representations of the input model for slicing, and approximating curvature using voxels. Despite providing good curvature approximation for systems well suited for voxel space like polyjet printing, their algorithm does not provide an efficient way to reduce line segments in traditional toolpath driven processes. Q. Li et al. expressed the infancy of the implicit slicing topic in their review [7]. They found that most existing CAD tools are focused on subtractive manufacturing and lack support for 3D capabilities. For instance, most existing implicit subtractive tools are concerned with boundary representations. This is not well suited for AM processes that use various densifying structures to reduce material use or improve strength. While a completely implicit slicer replacement is widely sought after, there are many challenges that need addressed before one will become widely available.

Given the current limitations of implicit slicing, there has been extensive investigation into fitting parametric curves to line segments [10, 12, 21, 23]. However, these papers have focused on traditional CNC path planning and there has been little work applying the techniques to AM. Valdivieso proposed a greedy algorithm to collect possible arc candidate locations and fit them to

line segments in AM [16]. Leveraging the G2 and G3 commands supported by most AM motion controllers, Valdivieso's work was primarily concerned with using biarcs to approximate curvature in an additively manufactured object. Most existing CNC toolpath planners rely on biarcs to approximate b-splines because they often lack support for the G5 command [3, 17, 22, 24]. Although this results in a reduction of motion commands over exclusively using line segments, it requires many arcs to represent dynamic curves. In contrast, graphical methods for curve fitting discrete data routinely employ splines [11, 15, 20]. Lin and Ding proposed a method to fit b-splines to ordered data points [9]. Similarly, W. Li et al. [8] presented an adaptive knot placement algorithm for b-spline curve approximation of discrete data.

Consequently, this paper focuses on providing implicit capabilities through post-process curve fitting. This approach leaves the traditional slicing process intact, allowing the user to leverage standard toolpath planning algorithms for cross-sectioning, infill generation, and support structure calculation. The proposed method takes a series of line segment commands as input and returns a combination of arcs, splines, and lines that form an implicit representation of the toolpath. Input toolpaths can be either closed or open polyline contours but must contain at least two segments to fit a curve. A post-processing approach provides a non-intrusive method to leverage the toolpath planning capabilities in explicit geometry slicers, while enabling implicit construction with arcs and splines.

Parametric Curve Fitting

Arcs

The G2 and G3 commands define arcs using three parameters: a start point, an end point, and a center point. G2 is used for clockwise motion, while G3 is used for counterclockwise motion. When fitting an arc to a pair of consecutive line segments, the center point of the arc that circumscribes the segments must be determined. Figure 2 illustrates how the center point can be computed from two segments by finding the intersection of their perpendicular bisectors.

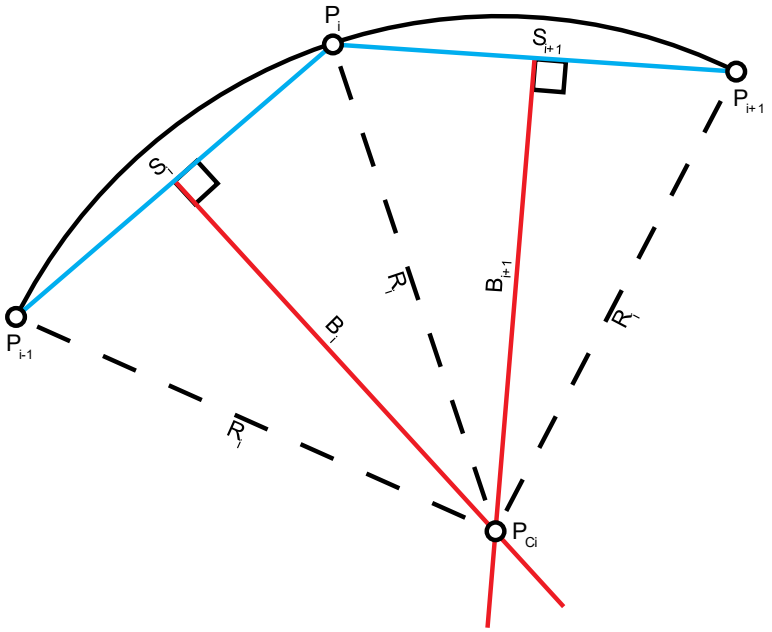


Figure 2: Finding the center point of an arc fits segments S_i and S_{i+1}

Given a series of N ordered line segments, it can be determined if their radii of curvature match. It is observed that consecutive line segments with identical curvature can be fit with a single arc. This is accomplished by picking a median point from the series and reducing the problem to a two-segment system. The center point can then be calculated for all points in the series. By partitioning a series of line segments into sections with matching curvature multiple arcs and lines can be fit optimally.

B-Splines

The G5 command uses four parameters to draw a b-spline: a start point, an end point, and two control points [13]. Although biarcs can approximate advanced curvature such as b-splines, they are less efficient than using the G5 command. Figure 3 shows that multiple G2 and G3 arc commands must be used to express the same corresponding G5 b-spline. The topic of fitting b-splines to discrete data points is the focus of multiple papers [8, 9]. To simplify this work and to focus on hybridized arc and spline fitting, the W. Li et al. approach is used to determine the placement of b-splines for a series of line segments [8]. The algorithm proposed by W. Li et al. takes an ordered series of points, representing the start and end points of the line segments, and returns the optimal knot placement to construct b-splines.

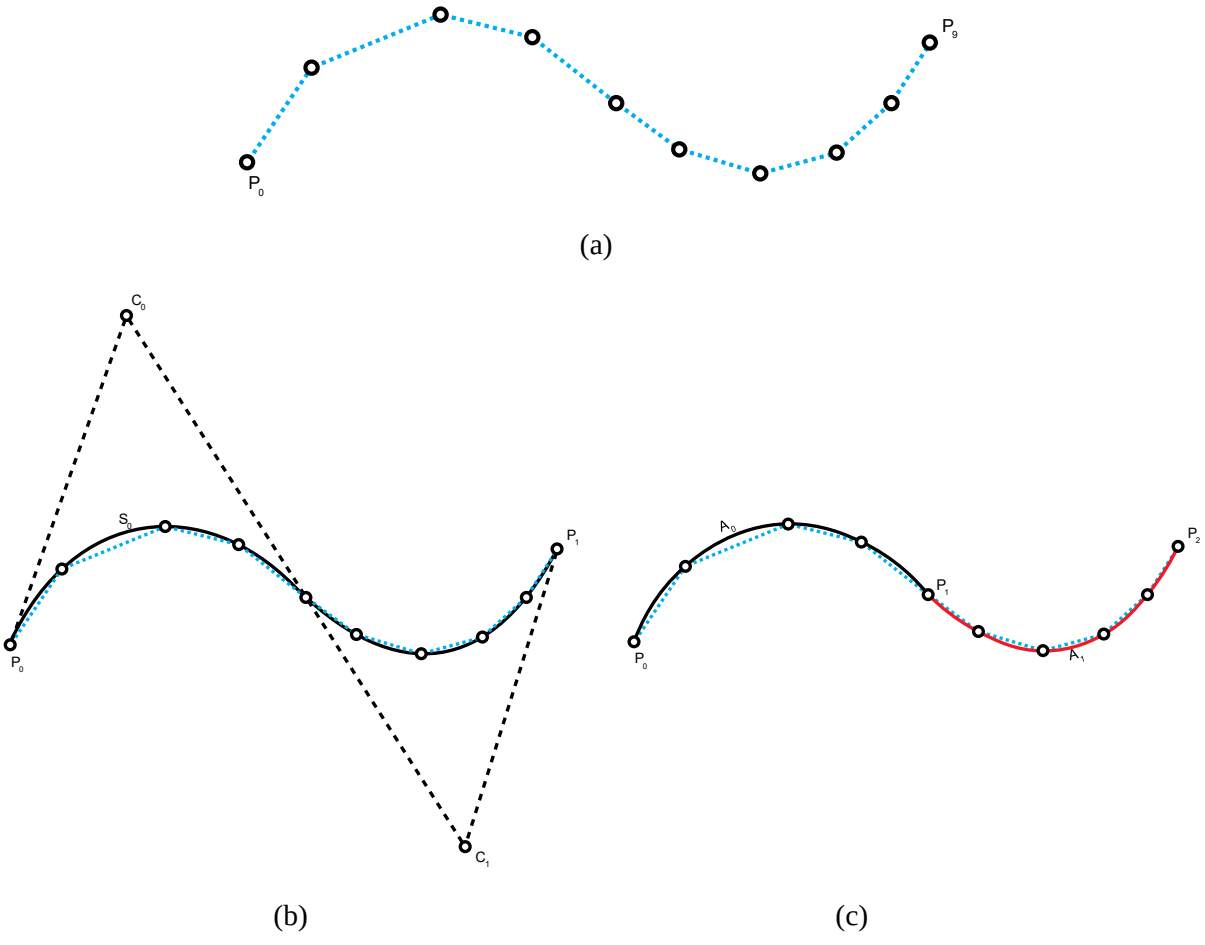


Figure 3: (a) series of line segments, (b) fit with a single G5 b-spline, (c) fit with one G2 and one G3 arc

Hybrid Fitting

Although certain geometries drawn through the G5 b-spline command are more concise, G5 is not a universal replacement for constant radius G2/G3 arcs. In the trivial example of Figure 4, the circle can be drawn with a single arc command, where two spline commands to achieve the same result are required. This exemplifies a need for a hybridized approach when fitting curves to line segments. The method for hybrid fitting first performs preprocessing to clean the input geometry. Next, it conducts initial partitioning based on an angle cutoff to preserve sharp contours. Then, curvature analysis is conducted to categorizes partitions based on their type. Finally, these sections are fit with the outlined arc and spline methods according to their category.

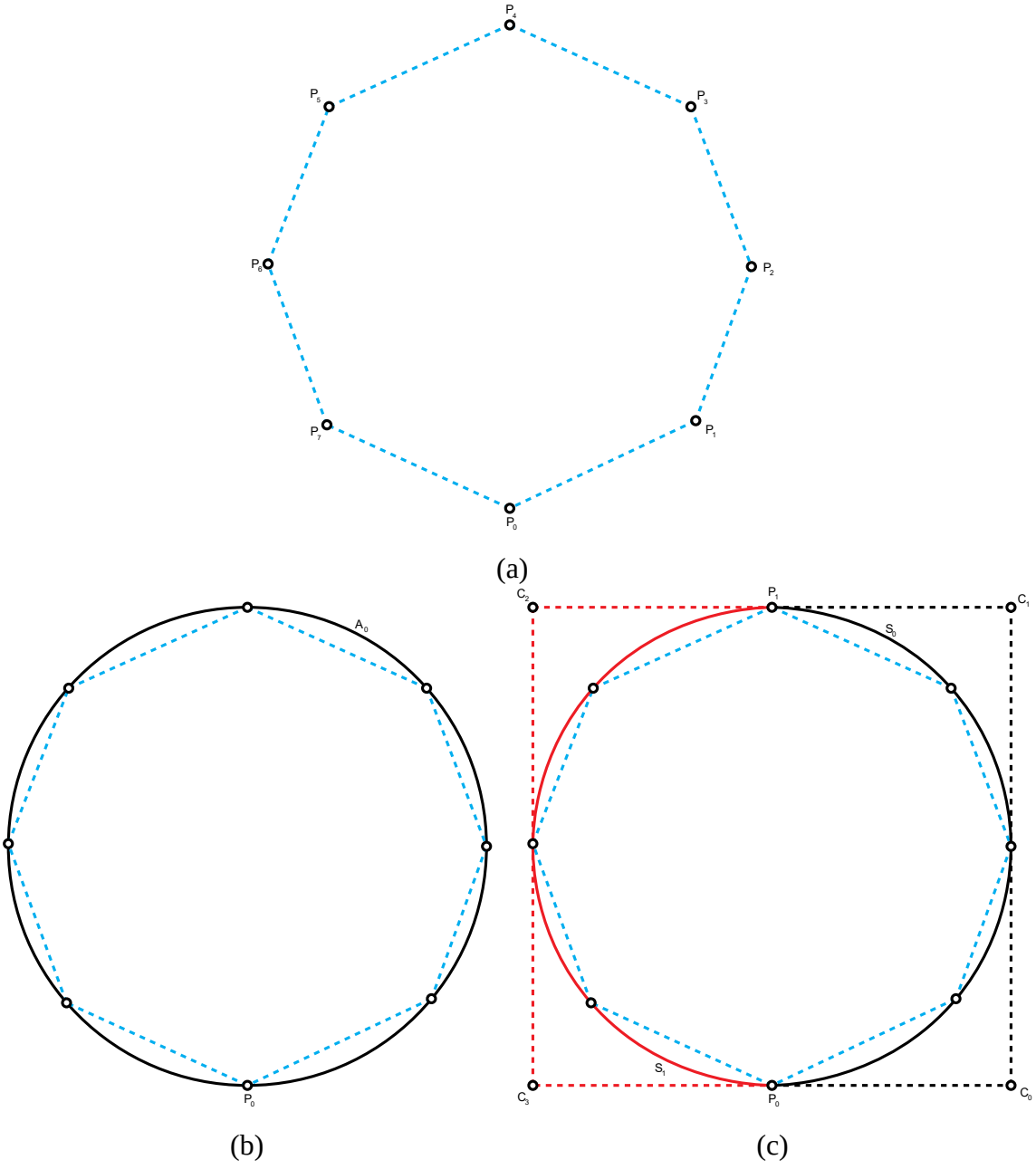


Figure 4: (a) Circular line segment path, (b) fit with a single arc, (c) fit with two splines

Preprocessing

Before a series of line segments can be fit with arcs and splines, it must be preprocessed to remove noisy and redundant data. Collinear segments do not contribute toward the hybrid fitting process and must be replaced with a single line segment. Likewise, segments of a very small length should be stitched with longer ones since they do not contribute substantially to the geometry. A linear smoothing parameter $\alpha = 10$ microns is used to identify sections of negligible length. However, this parameter depends on the scale of the machine. This work assumes that an input triangulated mesh was exported from CAD software with a reasonably high resolution. This assumption excludes models that were reconstructed from 3D scan or LIDAR data.

Initial Partitioning

The initial partitioning phase identifies subsets of line segments that may be replaced with parametric curves. First, pairs of line segments that form sharp angles are detected. If fit with a curve, these pairs would result in over-smoothing of the path. Given the ordered list of segments, $\{S_i, (i = 0, \dots, n)\}$, where n is the total number of segments, consider each pair $(S_i,\ S_{i+1})$. If the angle between segments, θ_i, falls below a critical value, φ, then the path is partitioned at the shared point, P_i. S_i remains in the current partition, while S_{i+1} starts a new one. Typically, a range of $\frac{\pi}{2} < \varphi < 2\pi$ is ideal for fitting curves, but this parameter must be tuned by the user.

The relationship between S_i and S_{i+1} and the angle θ_i is graphically shown in Figure 5, and can be expresses mathematically as:

$$\theta_i = \cos^{-1} \frac{\vec{S}_i \cdot \vec{S}_{i+1}}{|\vec{S}_i|\,|\vec{S}_{i+1}|} \qquad \text{(Eq. 1)}$$

where, $\vec{S}_i = P_{i-1} - P_i$ and $\vec{S}_{i+1} = P_{i+1} - P_i$.

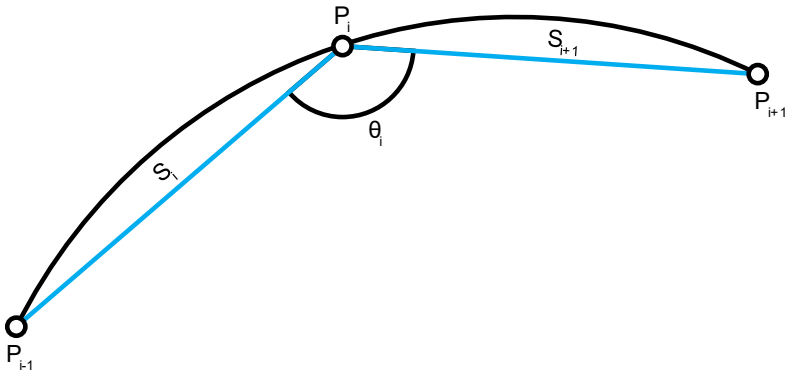


Figure 5: Calculating θ_i from segments

Curvature Partitioning

Initial partitioning results in several sets of line segments that need to be fit with parametric curves. The curvature partitioning phase is the critical step of determining which line segments are best fit with arcs or splines. If a given partition contains a single line segment, no curve will be fit. The nature of parametric curves enables subsections to be classified based on the change in the discrete signed curvature of pairs of segments. Intuitively, discrete signed curvature describes how quickly consecutive line segments are changing their angle.

Discrete signed curvature can be expressed mathematically as:

$$k_i = sign(\tau_i)\frac{1}{R_i} \quad \text{(Eq. 2)}$$

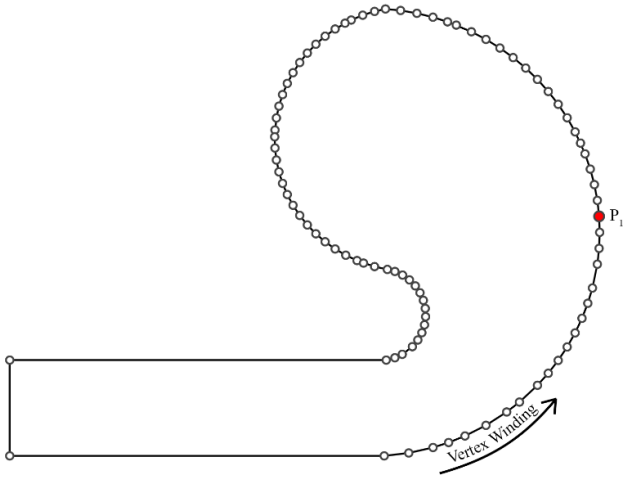
where, $\tau_i = \vec{S}_i \times \vec{S}_{i+1}$ as a scalar, $R_i = |P_{Ci} - P_i|$, and S_i, S_{i+1} are not parallel.

The definition of discrete signed curvature is rooted in finding the inverse radius R_i of a circle that passes through a set of three points. The signed component can be used to detect whether curvature is clockwise or counterclockwise. It can be surmised that when curvature remains constant across a set of consecutive line segments, they are best fit with an arc. Conversely, when the curvature is dynamic, it is best fit with b-splines. Using this principle, subsections that are best fit with different curvature types are identified:

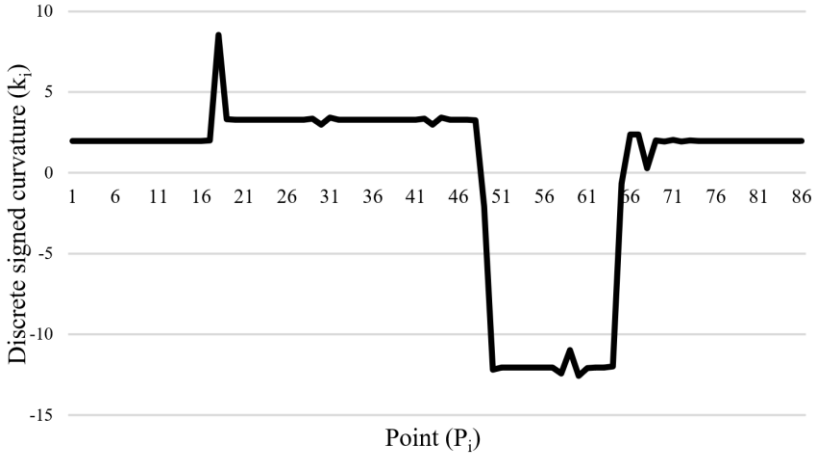
Given a set of n ordered segments $\{S_i, (i = 0, \dots, n)\}$ where $n > 1$, each pair can be classified based on the change in their curvature $\Delta k = k_{i+1} - k_i$.

$$type = \begin{cases} arc, & \Delta k \cong 0 \\ spline, & otherwise \end{cases} \quad \text{(Eq. 3)}$$

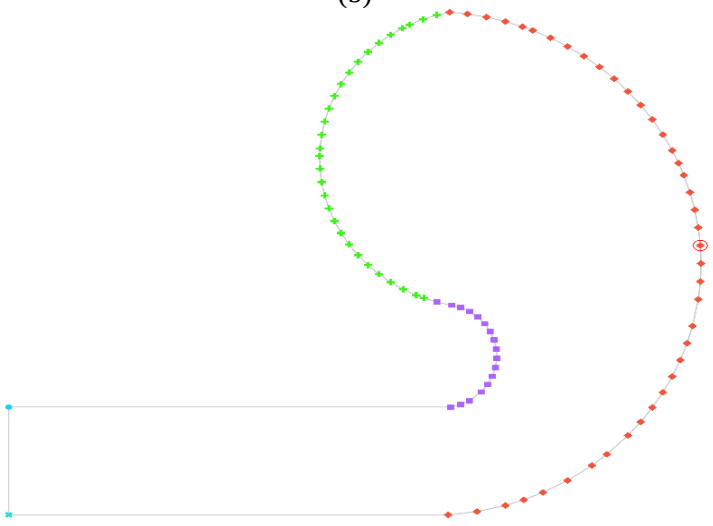
Graphically, the relationship between curve type and discrete signed curvature can be seen clearly. The examples below are cross-sections of triangulated meshes taking during the slicing process. The 3D models were exported from an implicit representation in modeling software to a triangulated mesh, discarding exact curvature information. When the cross-section is taken, a series of line segments is generated approximating the surface of the mesh. Figure 6(a) shows the cross-section of a 3D model that was designed with three arcs and three line segments. Line segments were identified using the angle analysis in the initial partition phase. Note in Figure 6(b) that sections of line segments whose curvature is constant can be fit with arcs. The grouped segment points expressed in Figure 6(c) demonstrate how the series was partitioned.



(a)



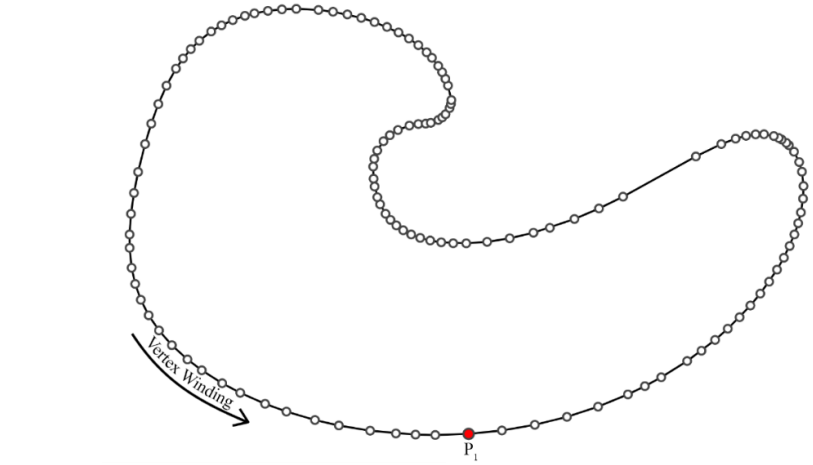
(b)



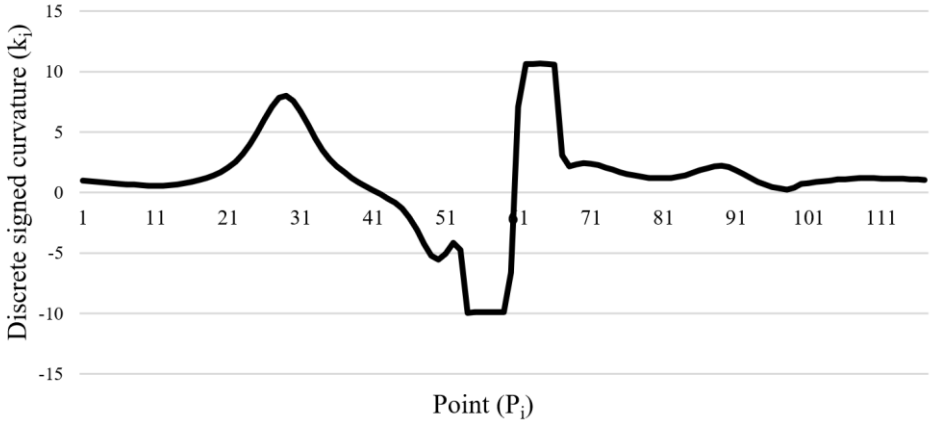
(c)

Figure 6: (a) Input model containing three arcs and three line segments, (b) discrete signed curvature of each consecutive pair of segments, (c) line segment points partitioned by curvature

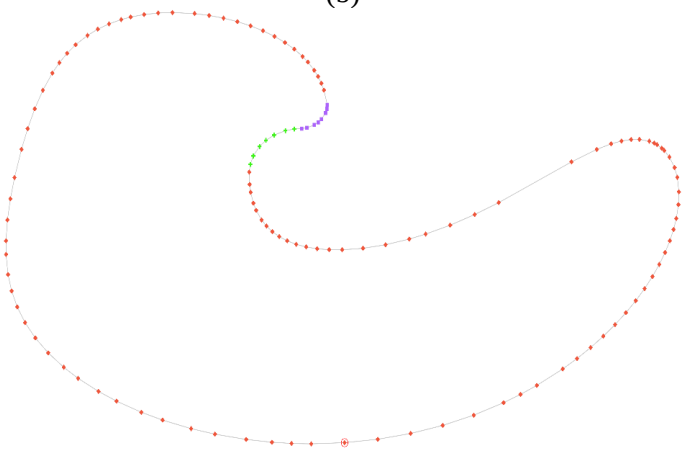
Similarly, Figure 7(a) depicts a model that is constructed from two arcs and a b-spline. When plotted in Figure 7(b), the two arcs are identified by grouping consecutive sections based on constant curvature. Since the other sections are dynamic, they are identified as a b-spline. Figure 7(c) demonstrates the results of the hybrid fitting algorithm as classified segment points.



(a)



(b)



(c)

Figure 7: (a) Input model containing a b-spline and two arcs, (b) discrete signed curvature of each consecutive pair of segments, (c) Line segment points partitioned by curvature

Even when preprocessing was applied, noise is still present in both above examples. This is the result of the Standard Triangle Language (STL) triangulation algorithm that is applied when models are exported from an implicit representation in CAD software. To compensate for noise in the discrete signed curvature data caused by triangulation, a threshold β is used to determine if there was a substantial change in curvature. Intuitively, β is the maximum deviation from the mean of the change in curvature plus or minus δ times the standard deviation of the change in curvature. This statistical cutoff provides a method for filtering the noise generated by triangulation, while maintaining the classification capabilities when performing discrete signed curvature analysis. The value $\delta = 2$ was used for this work; however, this parameter may need to be tuned for noisier datasets.

Mathematically, β is expressed as:

$$\beta = \mu_{\Delta k} \pm \delta\sigma_{\Delta k} \quad \text{(Eq. 4)}$$

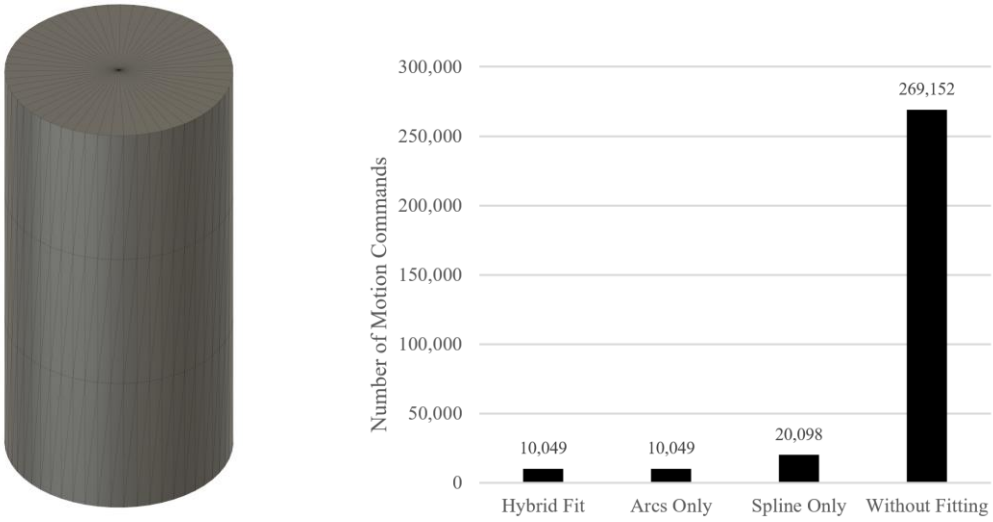
where, $\mu_{\Delta k} = \frac{1}{n}\sum_{i=1}^{n} \Delta k_i$ and $\sigma_{\Delta k} = \sqrt{\frac{\sum(\Delta k_i - \mu_{\Delta k})}{n}}$

Results

In this section, we present several examples whose parametric curves were reconstructed from line segments using the proposed algorithm. The algorithm was implemented as a post-processing step into the ORNL Slicer 2.0 program [1]. This allowed for objects to be constructed with skins, shells, and a concentric infill pattern. Although the algorithm can be used with any set of polylines, concentric infill is the only pattern that will result in arcs or spline fits in Slicer 2. If selected, other infill patterns such as triangles or hexagons will not be fit with arcs or splines because their internal angles are below the threshold φ. ORNL Slicer 2.0 additionally provided cross-sectioning and g-code exporting capabilities. Each model was designed with curvature in CAD software, exported as a triangulated mesh, cross-sectioned into layers, and then fit with arcs and splines. An analysis is provided into the toolpath file size reduction. Values of $\varphi = \frac{3\pi}{4}$, $\delta = 2$, and $\alpha = 10$ were used.

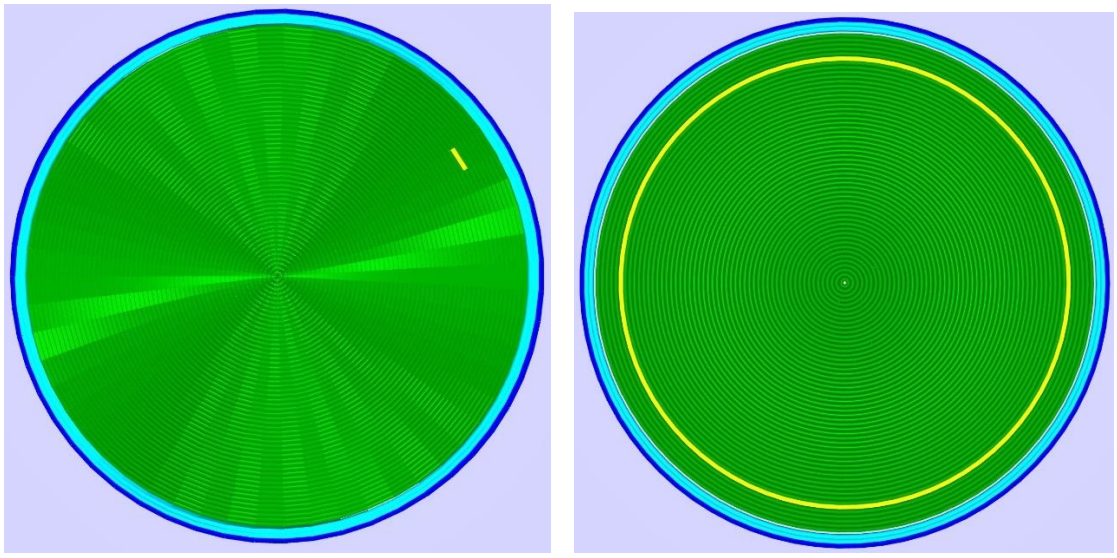
Example 1: A Cylinder

The model in Figure 8(a) is a cylinder with a radius of 25 mm and a height of 100 mm. Given the nature of a cylinder, the model contains one circle per cross-section. From the toolpaths shown in Figure 9(a), it is observed that the cross-sectional outline is offset numerous times to form insets and skins. This results in multiple circles that need fit with G2 and G3 commands per layer. The arc-fit pathing in Figure 9(b) shows how each circle can be drawn with a single arc command. As indicated by Figure 8(a), the application of hybrid and arc-only fitting resulted in a reduction of motion commands by a factor of over 26 when compared with using exclusively line segments. Similarly, the application of spline-only fitting resulted in a reduction by a factor of 13. This epitomizes the situation where arc-only fitting is superior to spline fitting. In this case, hybrid fitting produces the same result as arc-only fitting.



(a) (b)

Figure 8: (a) input cylinder model, (b) number of motion commands required to construct the object

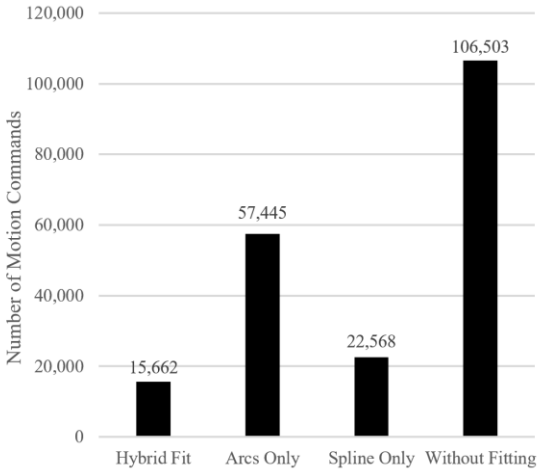


(a) (b)

Figure 9: (a) toolpaths using line segments only, (b) toolpaths after hybrid fitting, yellow sections show a single segment.

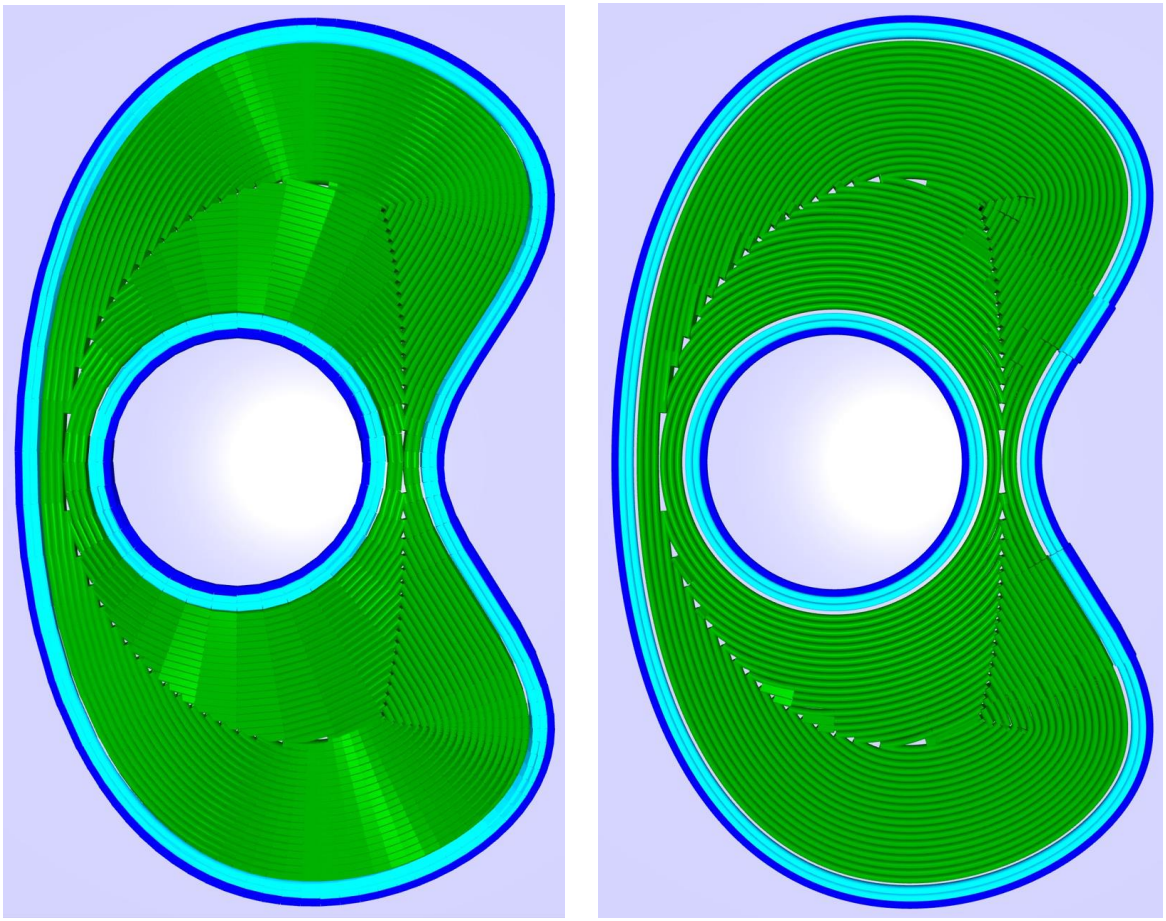
Example 2: NURBS with Hole

Seen in Figure 10(a), this example is a model designed with a continuous non-uniform rational basis spline (NURBS) around the outside, and a circular hole removed from inside. In Figure 11(a), the cross-sectioned NURBS curve results in a series of line segments best fit with b-splines. The circular hole will result in line segments that are best fit with a single arc command. Using the proposed method, the b-spline and arc segments were identified correctly and fit using their corresponding g-code commands. A major benefit of hybrid fitting can be realized in figure 10(b) by comparing the total number of motion commands required to construct an object. Arc fitting on this model provides a modest reduction in motion commands by a factor of 1.9 when compared with only line segments. Spline fitting improves this to a factor of 4.7 and hybrid fitting performs the best with a reduction of 6.8. This model highlights the benefit of hybrid fitting over single-curve-type based approaches.



(a) (b)

Figure 10: (a) input model drawn from a NURBS surface and circular hole, (b) number of commands required to construct the object

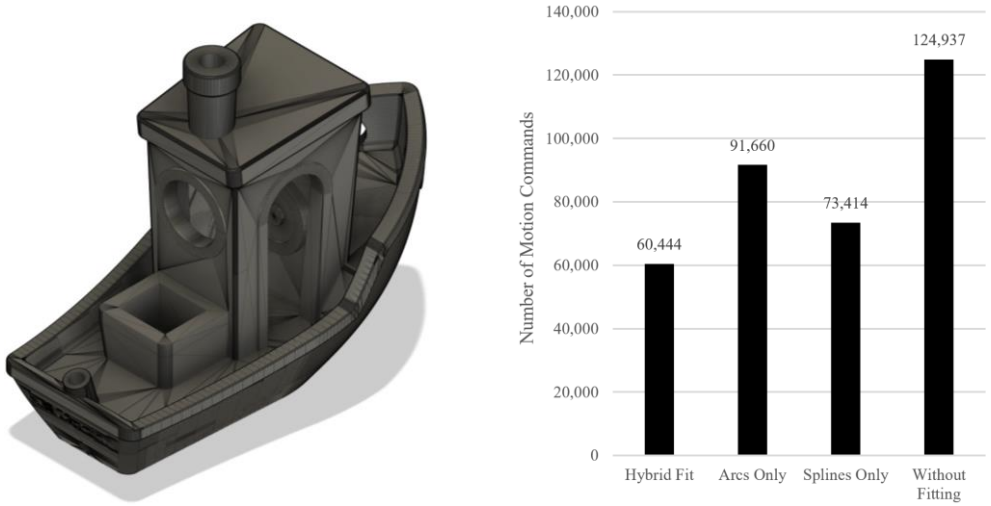


(a) (b)

Figure 11: (a) toolpaths using line segments only, (b) toolpaths after hybrid fitting

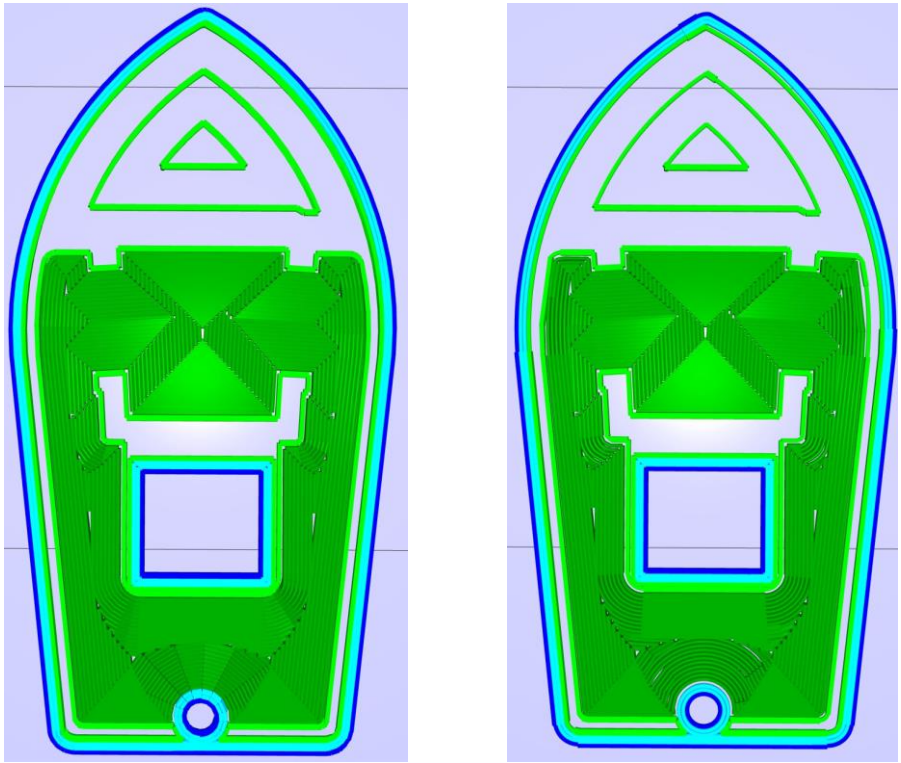
Example 3: 3D Benchy

Figure 12 is the popular AM benchmarking model: 3DBenchy.stl. The boat is comprised of multiple sections that can be hybrid fit to reduce motion commands. Figure 13 shows a layer of toolpaths before and after hybrid fitting. Notable sections include: the bow section was replaced with splines, the small circle on the stern was replaced with circle arc commands, and several rounded corners were replaced with arcs. Using hybrid fitting cut the number of motion commands needed to construct the boat by 50% compared to only using line segments. Arc-only and spline-only fitting resulting in a reduction in commands by 27% and 41% respectively.



(a) (b)

Figure 12: (a) input 3DBenchy.stl, (b) number of commands required to construct the object



(a) (b)

Figure 13: (a) object built with only line segments, (b) object build with hybrid fit

Impact on Slicing Time

In addition to recording the total number of toolpaths for each case study, wall-clock time was also measured for each case study. Table 1 demonstrates that the proposed hybrid curve fitting algorithm does not majorly impact the time required to slice and visualize an object. In the Cylinder case study, the tool-path generation phase was slower due the added overhead of running the algorithm. However, the reduction of motions commands after this phase compared with no fitting results in faster optimize, write G-Code, and visualization steps. This contributes to hybrid fitting having a faster total slicing time when compared with no fitting. In the NURBS case study, a similar relationship is observed. In this case the total times are nearly identical. The hybrid fitting on the 3DBenchy ends up marginally slowing the slicing process. These examples suggest a correlation between the number of line segments replaced with curvature and a speed-up in processing time. This leads to the conclusion that even in cases where there is a modest reduction in motion commands, the algorithm presents only a marginal impact on slice time.

Model	Fitting	Tool-path Generate (s)	Optimize (s)	Write G-Code (s)	Visualize (s)	**Total (s)**
Cylinder	*None*	1.3	0.6	5.2	3.0	**10.0**
	Hybrid	7.1	0.2	1.0	0.5	**9.0**
NURBS	*None*	0.9	0.1	1.6	0.9	**3.5**
	Hybrid	3.0	0.1	0.3	0.2	**3.6**
3DBenchy	*None*	3.1	0.3	2.4	1.6	**7.4**
	Hybrid	6.2	0.3	1.3	0.9	**8.6**

Table 1: Wall-clock time of each phase in the slicing workflow for case studies using no fitting and hybrid fitting

Conclusion and Further Work

A hybrid method for arc and spline reconstruction is presented in this paper for AM toolpath planning. It can dramatically reduce the number of required motion commands to construct an object with curved surfaces. This reduction results in smaller g-code files and a better surface appearance of additively manufactured parts. Combining dense sections of line segments reduces choppy motion, thereby enabling machines with limited computational kinematic throughput to print more complex objects. Furthermore, we demonstrated that the proposed hybrid fitting can provide a tangible reduction in the number of motion commands over arc- and spline-only approaches to the parametric curve fitting problem. Leveraging discrete signed curvature analysis for hybrid fitting provides a robust method for classifying line segments in paths with dynamic geometry. Applying this method as a post-processing step enables seamless integration into existing slicing workflows, avoiding the complexities of developing a purely implicit toolpath planning program.

Although the hybrid method is tolerant towards models with minimal noise, further investigation into fitting triangulated meshes that are reconstructed from noisy scan data is needed. W. Li et al. proposed using a low-pass filter to preprocess noisy scan data with their adaptive b-spline knot placement algorithm [8]. Perhaps, the same principle could be applied to noisy, reconstructed meshes. The outlined method also performs better when the input mesh is

triangulated with a uniformly high refinement. More investigation is needed into data preprocessing to provide better accuracy with paths that are sparsely populated. Although the proposed method enables the implicit representation of printed geometry, it still relies on approximating CAD models with triangulated meshes as an intermediate step. This introduces the potential for slight deviations in printed object geometry when compared with the purely implicit CAD representation. The benefits of using the algorithm with curve-based infill such as the gyroid pattern could also be investigated. Finally, as more implicit geometry AM tools are developed, this tool could provide an interesting method for converting existing toolpaths into solid implicitly represented parts.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, Office of Advanced Manufacturing, under contract number DE-AC05-00OR22725.

References

1. Borish, Michael, and Alex Roschli. "ORNL Slicer 2.0: Towards a New Slicing Paradigm." 2021 International Solid Freeform Fabrication Symposium. University of Texas at Austin, 2021.
2. Brunton, Alan, and Lubna Abu Rmaileh. "Displaced signed distance fields for additive manufacturing." ACM Transactions on Graphics (TOG) 40.4 (2021): 1-13.
3. Drysdale, RL Scot, Günter Rote, and Astrid Sturm. "Approximation of an open polygonal curve with a minimum number of circular arcs and biarcs." *Computational Geometry* 41.1-2 (2008): 31-47.
4. Hong, Qingqi, et al. "A direct slicing technique for the 3D printing of implicitly represented medical models." *Computers in Biology and Medicine* 135 (2021): 104534.
5. Huang, Pu, Charlie CL Wang, and Yong Chen. "Intersection-free and topologically faithful slicing of implicit solid." Journal of computing and information science in engineering 13.2 (2013).
6. Jin, Yuan, et al. "An optimization approach for path planning of high-quality and uniform additive manufacturing." The International Journal of Advanced Manufacturing Technology 92.1 (2017): 651-662.
7. Li, Qingde, et al. "Towards additive manufacturing oriented geometric modeling using implicit functions." *Visual Computing for Industry, Biomedicine, and Art* 1.1 (2018): 1-16.
8. Li, Weishi, et al. "Adaptive knot placement in B-spline curve approximation." *Computer-Aided Design* 37.8 (2005): 791-797.
9. Lin, Zizhi, and Yun Ding. "B-Spline Curve Fitting with Normal Constrains in Computer Aided Geometric Designed." *International conference on Big Data Analytics for Cyber-Physical-Systems*. Springer, Singapore, 2020.
10. Moreton, D. N., D. B. Parkinson, and W. K. Wu. "The application of a biarc technique in CNC machining." *Computer-aided engineering journal* 8.2 (1991): 54-60.
11. Park, Hyungjun, and Joo-Haeng Lee. "B-spline curve fitting based on adaptive curve refinement using dominant points." Computer-Aided Design 39.6 (2007): 439-451.
12. Parkinson, D. B., and D. N. Moreton. "Optimal biarc-curve fitting." *Computer-Aided Design* 23.6 (1991): 411-419.
13. RepRap Project. “Bézier Cubic Spline.” Marlin G-Code Reference

14. Roschli, Alex, et al. "Designing for big area additive manufacturing." Additive Manufacturing 25 (2019): 275-285.
15. Ruppert, David, and Raymond J. Carroll. "Theory & methods: Spatially-adaptive penalties for spline fitting." Australian & New Zealand Journal of Statistics 42.2 (2000): 205-223.
16. Salmerón Valdivieso, Honorio. "Polyline defined NC trajectories parametrization. A compact analysis and solution focused on 3D Printing." *arXiv e-prints* (2018): arXiv-1808.
17. Schönherr, Jens. "Smooth biarc curves." *Computer-Aided Design* 25.6 (1993): 365-370.
18. Song, Yanzhi, et al. "Function representation based slicer for 3D printing." *Computer Aided Geometric Design* 62 (2018): 276-293.
19. Sutherland, Jeffrey J., Lee A. O'brien, and Donald F. Weaver. "Spline-fitting with a genetic algorithm: A method for developing classification structure− activity relationships." Journal of chemical information and computer sciences 43.6 (2003): 1906-1915.
20. Steuben, John C., Athanasios P. Iliopoulos, and John G. Michopoulos. "Implicit slicing for functionally tailored additive manufacturing." *Computer-Aided Design* 77 (2016): 107-119.
21. Tseng, Y-J., Y-D. Chen, and C-C. Liu. "Numerically controlled machining of freeform curves using biarc approximation." *The International Journal of Advanced Manufacturing Technology* 17.11 (2001): 783-790.
22. Yang, Xujing, and Zezhong C. Chen. "A practicable approach to G1 biarc approximations for making accurate, smooth and non-gouged profile features in CNC contouring." *Computer-Aided Design* 38.11 (2006): 1205-1213.
23. Yeung, Millan K., and Desmond J. Walton. "Curve fitting with arc splines for NC toolpath generation." *Computer-Aided Design* 26.11 (1994): 845-849.
24. Yuan, En Tao, and Bing Shao. "Biarc Approximations Tool-path Generation for Polyhedral Models." *Advanced Materials Research*. Vol. 889. Trans Tech Publications Ltd, 2014.