

Automated Path Planning for Wire Feeding in Large Format Polymer Additive Manufacturing

Michael Borish*, Alex Roschli*

*Manufacturing Science Division, Oak Ridge National Laboratory, Knoxville, TN 37932

Abstract

Polymer-based large format industrial additive manufacturing (AM) technology continues to expand into new application areas. One area of interest is large scale composite molds and dies. These molds and dies can be used for out-of-autoclave tooling applications. However, at these sizes, several challenges remain that prevent the use of AM technology due to cost. One such challenge is the need to heat these molds in large thermal ovens. To address this challenge, researchers at Oak Ridge National Laboratory developed the necessary hardware to allow co-extruded wire to be embedded into the material during construction. Using this hardware, a demonstration mold was successfully constructed and subjected to mechanical testing. The construction of this object required a unique pathing solution to achieve success. In this paper, we describe the needed software development in ORNL Slicer 2.0 to allow the automated production of this unique pathing solution.

Keywords: slicing, additive manufacturing, wire coextrusion, wire embedding, self-heating molds, composite molds

Introduction

Polymer-based large format AM technology has proven attractive due to the large scale and speed with which objects can be constructed. The pellet fed systems typically use fiber reinforced polymer composites that have proven useful to multiple industries including aerospace, automotive, and renewable energy [1-3]. One such system shown in Figure 1 is Big Area Additive Manufacturing (BAAM) [4]. BAAM has successfully constructed molds for tooling applications [5]. Large format polymer printers like BAAM have successfully



Figure 1: Big Area Additive Manufacturing (BAAM). Example large-scale gantry-based pellet fed polymer system.

This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (<http://energy.gov/downloads/doe-public-access-plan>).

demonstrated a significant reduction in the amount of time and cost of mold fabrication. These molds have been used in out-of-autoclave applications as an alternative to traditional autoclave approaches.

Traditional autoclave approaches allow the construction of high-performance composite objects. The process generally involves material laid upon a rigid mold that is then heated in an oven under pressure. Despite the success of this process, it is cost intensive in terms of capital, labor, and time [6]. To address some of these issues, out-of-autoclave (OOA) approaches were developed. OOA generally involves vacuum-assisted consolidation with several distinct steps including: the construction of the mold, laying of material, void removal via vacuuming, curing in a thermal oven, post-cure heating, and removal [7]. OOA has been successful in general, but the OOA process still relies on the use of a thermal oven to cure the resin. Naturally, as molds scale in size, the cost of purchase and operation of such a large oven becomes an issue.

One way to address limitations in the OOA process is the production of molds with embedded heating elements. This construction is possible via coextrusion of a wire filament with a polymer additive process. Coextrusion is not limited to molds. There are numerous applications both small- and large-scale applications as well as various combinations of build process and coextruded material. Coextruded wire or liquid metal has been used for the fabrication of electronics and stretchable circuitry [8-9]. Coextrusion is also not limited to a specific material type as continuous fiber for polymer composites, ceramics, and concretes has been investigated as well [10-14].

Indeed, researchers at Oak Ridge National Laboratory have investigated coextrusion with the BAAM system in the past. While a full description of the hardware used is outside the scope of this paper, details can be found in previous work [15, 16]. An example of the hardware is shown in Figure 2.

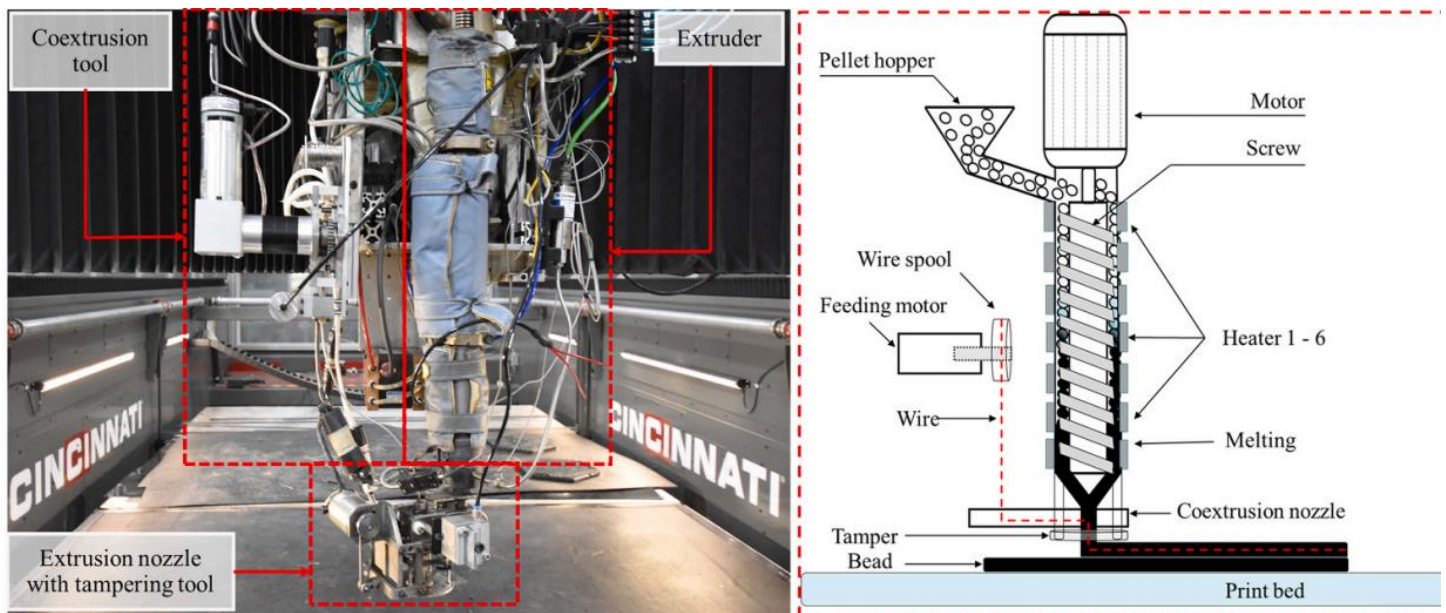


Figure 2: Co-extrusion hardware. A custom wire feed system is attached to the nozzle of the BAAM system [15].

In brief, the coextrusion system is comprised of a spool for material, a Midwest Motion Products DC motor and gearbox, a custom wire feed system, a wire feeding tube, a wire-cutter with Bimba air-cylinder, and a dual-port nozzle. This hardware was utilized as proof of concept to construct self-heated molds on the BAAM system [17, 18]. While the proof of concept was successful and other examples abound, this work and others relied on a very manual pathing process. That is, pathing modifications necessary to correctly place the co-extruded wire were developed by hand, and this pathing was developed for relatively simple geometries. Such a process is time consuming and is not practical for more complicated geometries.

To address these manual limitations in the coextrusion process, researchers at Oak Ridge National Laboratory developed a methodology to construct molds with embedded heating elements automatically. To

produce such an object, a unique pathing solution was required. This work focuses on the development of a unique slicing approach integrated into ORNL Slicer 2.0. This algorithm allows the automatic generation of pathing necessary to construct these molds as well as features necessary for wire coextrusion. In this paper, we outline these algorithms and provide a high-level overview of a produced demonstration piece and physical results.

Algorithm Design and Implementation

To provide an automated solution for the necessary pathing for wire feeding, ORNL Slicer 2.0 was modified with necessary functionality. To help visualize the goal of this algorithm, a representative rendering of the demonstration object with embedded wire is shown in Figure 3.

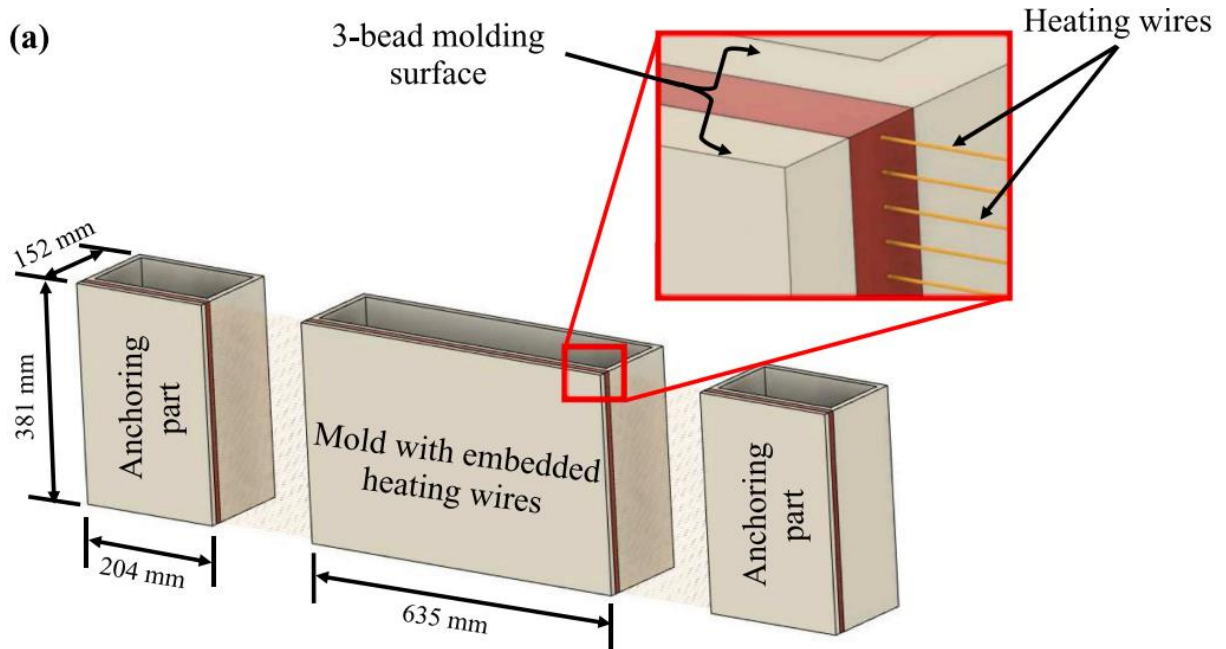


Figure 3: Rendering of mold with embedded heating element [17].

As can be seen from this figure, the mold with embedded wire is three beads in thickness. The wire embedded bead sits between the surface bead and the remaining geometry to ensure proper adhesion. For ease of discussion, the outer bead, wire embedded bead, and remaining geometry will be referred to as the surface region, wire region, and base region. Because the wire region must extend across the entire surface of the mold, pathing for the base region must be modified. Additionally, to ensure that the wire is taut during construction, sacrificial anchors are constructed on either side of this object. The purpose of the anchors is to provide a surface upon which the wire region can adhere as layer-wise construction continues. The wire region extends across the front of the mold to the outer boundary of each anchor.

To affect the necessary pathing solution, several significant components were developed in ORNL Slicer 2.0. These included settings mesh Boolean operations, anchor generation, unique base region pathing solution, and a restriction on post-processing travel optimization. The first feature that required expansion was the settings mesh. Slicer 2 allows a user to import multiple types of meshes. The most common mesh is the build mesh. This mesh represents the typical object in most slicers. Slicer 2 will cross-section this object and produce all necessary pathing based on user settings. In comparison, the settings mesh allows a user to modify existing pathing within a build mesh. An example of this is shown in Figure 4.

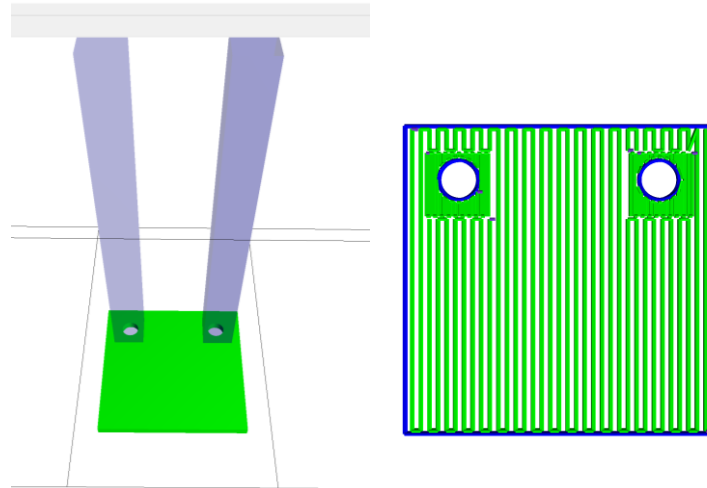


Figure 4: Example use of settings regions to affect pathing in specific locations of build mesh.

In this example, two settings meshes (transparent blue) were loaded on top of the square build mesh (green). In each of those settings meshes, the infill density was changed. As a result, the area around the holes were densified to provide additional stock should the holes be bored. Settings meshes as presented in Figure 4 were limited to modifying pathing in a specific region. In this case, infill and its relative density. Settings meshes were extended to provide Boolean mesh operations such as union and intersection of 3D bodies. The Boolean mesh operations were provided as part of Computational Geometry Algorithms Library (CGAL) [19]. A full explanation of 3D mesh intersection is outside the scope of this paper. However, CGAL is one well-known computational geometry library that provides detailed explanations of provided algorithms. These Boolean operations are applied before the pathing of the object is computed. An example is shown in Figure 5.

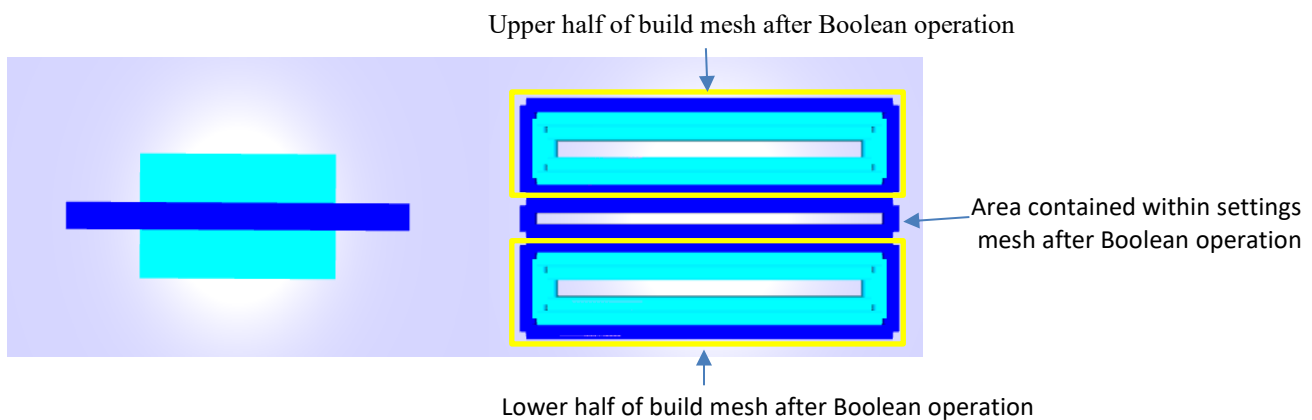


Figure 5: Settings mesh performed Boolean intersection of mesh before pathing computation.

In this case, the settings mesh (dark blue) performed a Boolean operation on the build mesh (light blue). Before cross-sectioning was applied, the settings mesh performed an intersection with the build mesh. The result from this intersection was the joined to the remaining geometry via mesh union. These geometric operations result three distinct islands for which pathing is computed. Because these regions have been split,

Slicer 2's pathing algorithms evaluate them separately. These three islands are the area above, below, and inside the settings mesh. This functionality was provided to allow a user to specify where in the object to embed wire along with other necessary user settings. Naturally, this process could also be affected by appropriate design via computer-aided design (CAD). If a user were to take the build mesh of Figure 5, split it into three separate STLs, and load those STLs into a slicing package, the same pathing could be produced. However, disassociating the various regions of interest into multiple independent STLs makes producing subsequent pathing more difficult. Rather, it is more straightforward to use the settings mesh to define an area of interest for wire coextrusion.

For subsequent explanation, the setup in Figure 6A will be used as a reference. This is a simple rectangular prism with a wire-embedded region specified by the settings mesh and represents the object shown in Figure 3. This setup is just before slicing has begun. With the settings mesh setup and appropriate user settings, slicing can begin. The first major step is cross-sectioning. As part of the cross-sectioning step, the build mesh was split according to the settings mesh. For this example, this process results in three areas of geometry: the surface region, wire region, and base region. A visualization of what the cross-section step processes is shown in Figure 6B.

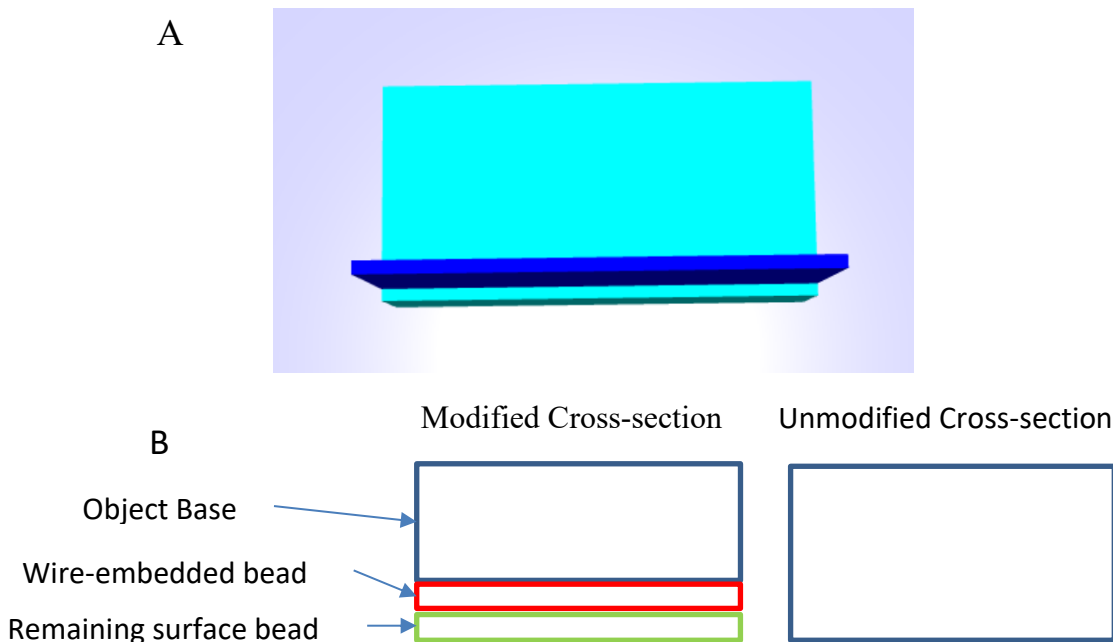


Figure 6:(A) Example mold (light blue) with settings region defining wire feed area (blue) just before slicing. (B) Cross-section output from first step in slicing process.

During the cross-sectioning step, two sets of cross-sections are computed. One cross-section is of the original geometry and one is of the geometry after settings mesh Boolean operations have been applied. Both sets of cross-sections are carried through the steps of the slicing process. The original cross-section will be used in combination with the modified cross-section to compute pathing for the base region of the object, while the modified cross-section will be used to produce pathing for the wire and surface regions. After the wire and surface region are removed and will be masked with the modified-cross section to manipulate the pathing. More details will be provided later in the section.

With cross-sectioning complete, one final pre-processing step must be accomplished before pathing computation can begin: anchor generation. For the wire in the wire region to remain taut during construction, the wire must be anchored on both sides of the object. These anchors are automatically generated based on user settings during the slicing process much like other programmatic features such as rafts, brims, or skirts. These anchors are simple squares to which an additional wire region can be affixed to provide the necessary tension

during the construction process. Unlike the mold object, these anchors do not have an additional surface region since they will not be machined. Once construction is complete, the wires will be cut and the anchors discarded.

Now that all appropriate geometry has been created, the slicing process can move to the path generation step. The pathing for the wire and surface region is computed using the modified-cross section. Both regions are typically one bead width wide. The pathing for these regions is straightforward and is the result of skeletons. Skeletons are open loop style paths that are typically used to fill in remaining voids inside of perimeters or other closed contours. In this case, the skeleton pathing produces a simple straight line for both regions.

No additional pathing is required for the surface region, however, the wire region requires one more addition. This region's pathing must be augmented to include pathing for the attachment points at the anchor. Like pathing already produced, these anchor points are single bead skeletons appropriately distanced based on user settings. Importantly, this pathing is bundled with the wire region such that it is handled as a combined unit. This has the practical effect of enforcing a purely increasing or decreasing build order once travels have been added at a later step.

Relatedly, the remaining pathing for the anchors is also computed. The anchors are programmatically generated based on user settings and are fundamentally like other features such as rafts. Users specify general settings for size and gap distance for example and the structures are generated. These anchors consist of a single perimeter bead with an additional skeleton bead along the front surface where the wire will be placed. An example of this pathing is shown in Figure 7.

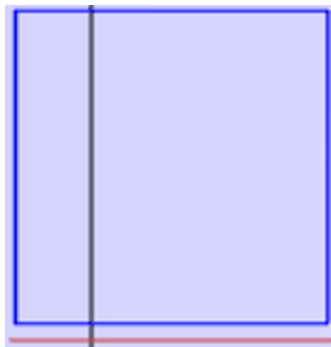


Figure 7: Anchor example pathing. Pathing consists of single perimeter bead (blue) and single skeleton bead (red) for wire.

The pathing for the base region is generated in parallel from the unmodified cross-section and is shown in Figure 8. Recall from the rendering in Figure 3, that the mold is three beads thick. The three beads are produced as a result of processing the unmodified cross-section as would happen in any normal slice. Once that pathing generation is complete, the outer two beads are altered. The outer two beads are intersected with the wire and surface regions from the modified cross-section. Polygon line-segment intersection can be decomposed into a simple line segment–line segment intersection algorithm. The Bentley-Ottmann algorithm is one such algorithm that allows computation of intersection points much faster than the naïve approach of evaluating every segment [20]. In brief, this algorithm is a line sweep algorithm that sweeps a line across all

segments that have been sorted by their x location for example. The resulting sweep produces a set of intersections that can be evaluated for path manipulation.

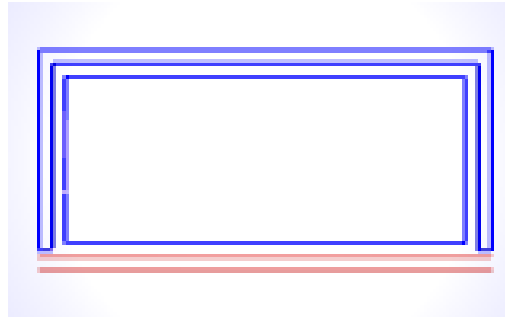


Figure 8: Example mold with modified pathing. Pathing consists of three modified perimeter beads (blue) and two skeletons (red): one for wire feed and one for the surface.

Using these intersections, the beads are split with any pathing falling inside the wire or surface regions discarded. The modified cross-section acts as a mask to determine what pathing has not already been computed as part of the other regions. The remaining pathing is no longer a closed contour as a result, so a merge process is applied. Every pair of contours is spatially sorted to identify the closest points to connect. Once these points are determined, the pathing is connected to once again form closed contours. Using this approach, an arbitrary number of beads is possible with each odd multiple allowing two preceding outer-most beads to be connected. In this case, the third bead is not altered as it does not intersect either of the other two regions.

With all this pathing laid out, the final step is a post-processing step to add travels. Typically, the pathing is connected via travels according to user settings without restriction. In this case, some restrictions were imposed. The pathing that represents the wire region must be aligned and connected last, always. Additionally, the pathing for this region must always travel from the minimum to maximum or vice versa to guarantee the wire is stretched across the entire object and anchored appropriately. An example slice with numbered pathing is shown in Figure 9.

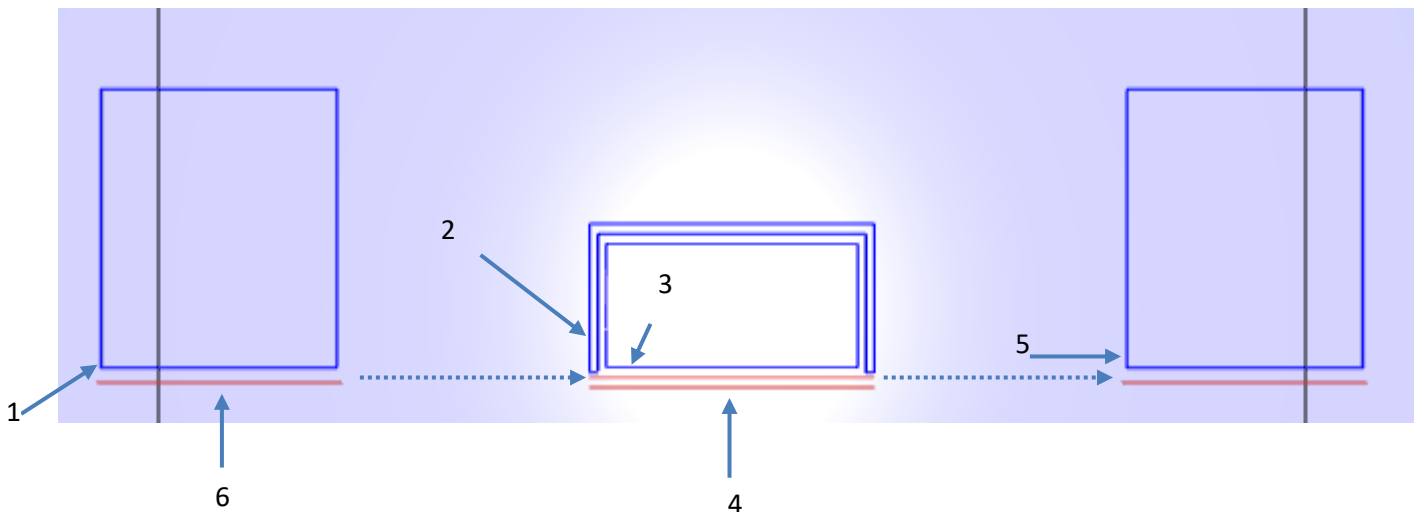


Figure 9: Example pathing of mold with anchors. Travels are inserted in numbered order to affect appropriate pathing.

As shown in this figure, the travel process begins at the contour bead of the left anchor (1), travels to the beads of the mold base (2-3), moves to the surface region (4), then moves to the right anchor (5), and finally constructs all three segments of the wire region (6). The travels insert with respect to all regions except for the wire region can be arbitrarily decided upon, however, the wire region must always be done last and from one extreme to another. With this pathing in place, molds with embedded heating elements can be produced automatically.

Demonstration and Discussion

Utilizing the generalized approach described in the previous section, a demonstration part was constructed. The part was constructed using the BAAM system with two different types of thermoplastic composite pellets. These two pellets were Polycarbonate with 20 wt.% carbon fiber loading (PC/CF) and polycarbonate with 20 wt.% glass fiber loading (PC/GF) (Techmer PM LLC (Clinton, TN, USA)). For the wire-embedded region, a nichrome alloy wire with Nickel 60%, Chromium 16%, and Iron 24%, and a diameter of 0.508 mm (McMaster-Carr, Cleveland, OH, USA) was used. The resulting object is shown in Figure 10.

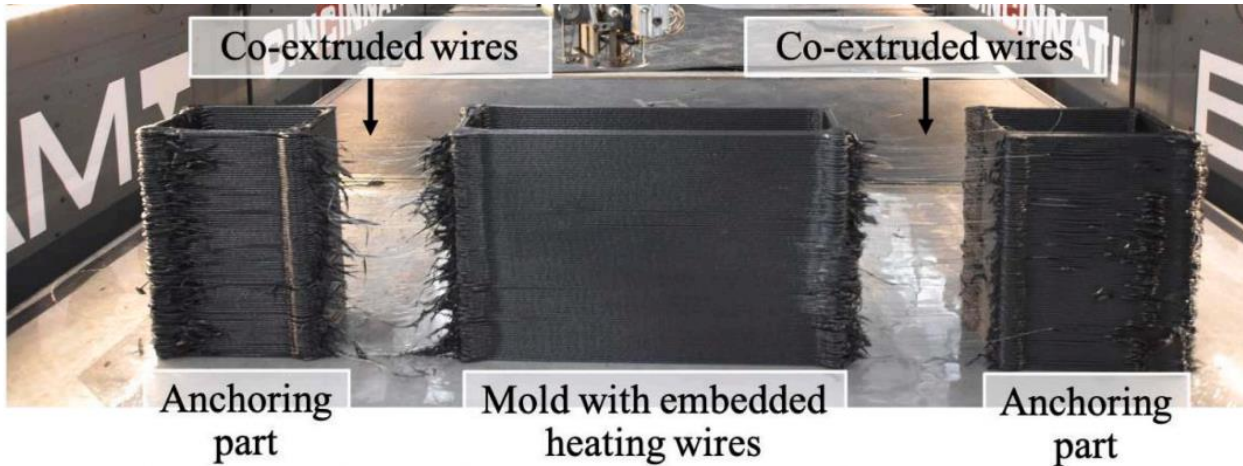


Figure 10: Demonstration mold with anchors and embedded heating elements [17].

A full discussion of the mechanical properties of the mold can be found in previous work [17]. In summary, the mold was subjected to tensile, flexural, and thermal testing. Results showed that the mold displayed reasonable tensile and flexural behavior and was able to achieve a uniform thermal distribution. This is important as one of the primary motivations of this work is the high cost of large ovens in the OOA process. To be clear, these mechanical results and demonstration object were accomplished using a manual pathing process. That is, the pathing was manually created and laid out by hand. However, this process can now be created automatically with the implementation previously described.

With respect to the algorithm presented in the previous section, the additional computation when compared to a typical slice is minor. Boolean mesh operations are accomplished via mesh refinement, and the operations are generally of linear complexity $O(n)$. So, the cross-section step of the slicing process takes about twice as long. The only other significant aspect to the process is the manipulation of the pathing for the mold's base region. The Bentley–Ottmann algorithm processes a sequence of $2n + k$ events, where n denotes the number of input line segments and k denotes the number of crossings. Each event requires $\log n$ time to evaluate. Ignoring constants, the complexity is thus $O((n + k) \log n)$. However, nearly all of the computational infrastructure is already in place due to the typical slicing process. Practically, this means that the additional computation adds only a few seconds to the typical slice.

Computational efficiency aside, the other practical benefit of this approach is the ability to support more complex geometries. While the implementation and examples to this point have been simple rectilinear geometries, this has been for ease of demonstration. There is nothing in this approach that requires rectilinear geometry. The algorithm as described is capable of ingesting meshes of any shape and complexity and producing pathing with the necessary commands for wire coextrusion. The only requirement is that a designer be able to produce a settings mesh representative of the volume in which wire coextrusion is expected to occur in.

Conclusion

As can be seen, the proposed algorithm requires only a minimal increase in computation to provide an automated solution. This pathing solution has been shown to allow the production of molds with embedded heating elements. Through physical testing, these molds were found to have appropriate tensile and flexural characteristics as well as uniform heating properties making them suitable for OOA applications. This construction was made possible by a wire coextrusion system attached to BAAM, a large format polymer platform. Though the presented approach has a number of advantages, there is still room for improvement. Future work will focus on the automated identification of the most appropriate region for wire coextrusion. Currently, this process requires the use of a second mesh, however, its possible that this need not be the case.

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.

Bibliography

- [1] K. Friedrich, A.A. Almajid, Manufacturing aspects of advanced polymer composites for automotive applications, *Appl. Compos. Mater.* 20 (2) (2013) 107–128, <https://doi.org/10.1007/s10443-012-9258-7>.
- [2] P.S. Veers, et al., Trends in the design, manufacture and evaluation of wind turbine blades, *Wind Energy* 6 (3) (2003) 245–259, <https://doi.org/10.1002/we.90>.
- [3] Polymer Composites in the Aerospace Industry, “Manufacturing Process. *Compos. Mater. Compon. Aerosp. Appl.*,” (2020) 59–81, <https://doi.org/10.1016/b978-0-08-102679-3.00003-4>.
- [4] Love, Lonnie, et al. Commercialization of Big Area Additive Manufacturing. No. ORNL/TM-2020/1454. Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), 2020.
- [5] A.A. Hassen et al., “The durability of large-scale additive manufacturing composite molds,” in *CAMX Conference Proceedings.*, September 26–29, 2016., 2016, no. April 2018, pp. 0–10.
- [6] D. Dixit, R. Pal, G. Kapoor, M. Stabenau, *Lightweight composite materials processing. Lightweight Ballistic Composites: Military and Law-Enforcement Applications*, Second edition., Elsevier Inc, 2016, pp. 157–216.
- [7] T. Centea, L.K. Grunenfelder, S.R. Nutt, A review of out-of-autoclave prepregs – material properties, process phenomena, and manufacturing considerations, in: *Composites Part A: Applied Science and Manufacturing*, 70, Elsevier Ltd, 2015, pp. 132–154, <https://doi.org/10.1016/j.compositesa.2014.09.029>.
- [8] Martinez, N. L. (2020). Development of an Automated Ultrasonic Wire Embedding Process for use with Material Extrusion Additive Manufacturing and Rapid Electronics Fabrication (Doctoral dissertation, The University of Texas at El Paso).
- [9] Khondoker, M. A., Ostashek, A., & Sameoto, D. (2019). Direct 3D Printing of Stretchable Circuits via Liquid Metal Co-Extrusion Within Thermoplastic Filaments. *Advanced Engineering Materials*, 21(7), 1900060.
- [10] Thakur, A., & Dong, X. (2020). Printing with 3D continuous carbon fiber multifunctional composites via UV-assisted coextrusion deposition. *Manufacturing letters*, 24, 1-5.
- [11] Garofalo, J., & Walczyk, D. (2018). In-situ Co-extrusion: additive manufacturing of continuous reinforced thermoplastic composites. In *Proceedings of the American Society for Composites—Thirty-third Technical Conference*.
- [12] Albrecht, H., Savandaiah, C., Lepschi, A., Löw-Baselli, B., & Haider, A. (2019). Parametric study in co-extrusion-based additive manufacturing of continuous fiber-reinforced plastic composites. In *Sim-AM 2019: II International Conference on Simulation for Additive Manufacturing* (pp. 417-427). CIMNE.
- [13] Jo, I. H., Koh, Y. H., & Kim, H. E. (2018). Coextrusion-based 3D plotting of ceramic pastes for porous calcium phosphate scaffolds comprised of hollow filaments. *Materials*, 11(6), 911.

- [14] Ducoulombier, N., Demont, L., Chateau, C., Bornert, M., & Caron, J. F. (2020). Additive Manufacturing of Anisotropic Concrete: a Flow-Based Pultrusion of Continuous Fibers in a Cementitious Matrix. *Procedia Manufacturing*, 47, 1070-1077.
- [15] Atkins, C., Heineman, J., Chesser, P., Roschli, A., Post, B., Lloyd, P., ... & Lind, R. (2019). Wire Co-Extrusion With Big Area Additive Manufacturing. Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States).
- [16] A. Roschli, K.T. Gaul, A.M. Boulger, B.K. Post, P.C. Chesser, L.J. Love, F. Blue, M. Borish, Designing for big area additive manufacturing, *Addit. Manuf.* 25 (September 2018) (2019) 275–285, <https://doi.org/10.1016/j.addma.2018.11.006>.
- [17] Billah, K. M. M., Heineman, J., Mhatre, P., Roschli, A., Post, B., Kumar, V., ... & Hassen, A. A. (2021). Large-scale additive manufacturing of self-heating molds. *Additive Manufacturing*, 47, 102282.
- [18] Billah, K. M. M., Hassen, A. A., Nasirov, A., Haye, G., Heineman, J., Kunc, V., & Kim, S. (2020, November). Thermal Analysis of Large Area Additive Manufacturing Resistance Heating Composites for Out of Oven/Autoclave Applications. In *ASME International Mechanical Engineering Congress and Exposition (Vol. 84485, p. V02AT02A040)*. American Society of Mechanical Engineers.
- [19] Giezeman, G.; Wesselink, W. 2D polygons. In *CGAL User and Reference Manual, 5.4 ed.*; CGAL Editorial Board, Eds.; 2022.
- [20] Bentley; Ottmann Algorithms for Reporting and Counting Geometric Intersections. *IEEE Trans. Comput.* 1979, C-28, 643–647. <https://doi.org/10.1109/tc.1979.1675432>.