Solid Freeform Fabrication 2017: Proceedings of the 28th Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference
Reviewed Paper

ORNL SLICER 2: A NOVEL APPROACH FOR ADDITIVE MANUFACTURING TOOL PATH PLANNING

Alex Roschli, Andrew Messing, Michael Borish, Brian K. Post, Lonnie J. Love

Manufacturing Demonstration Facility, Oak Ridge National Laboratory, Knoxville, TN 37932, USA

Abstract

ORNL Slicer is the first software designed to generate machine instructions, or tool paths, from CAD files for large-scale 3D printing of metals and polymers. The software was revolutionary because it allowed for slicing of models reaching 20 feet long, generating millions of lines of G-Code in seconds. The structure of the first ORNL Slicer had limitations in its framework, which has led to the development of ORNL Slicer 2. In the second version of the slicer, the process is modularized with individual layers being divided into regions, smarter infill patterns, and traversals are generated based upon stress, thermal, and other models. The new software has also been structured to allow for slicing and reslicing based on machine feedback during the printing process.

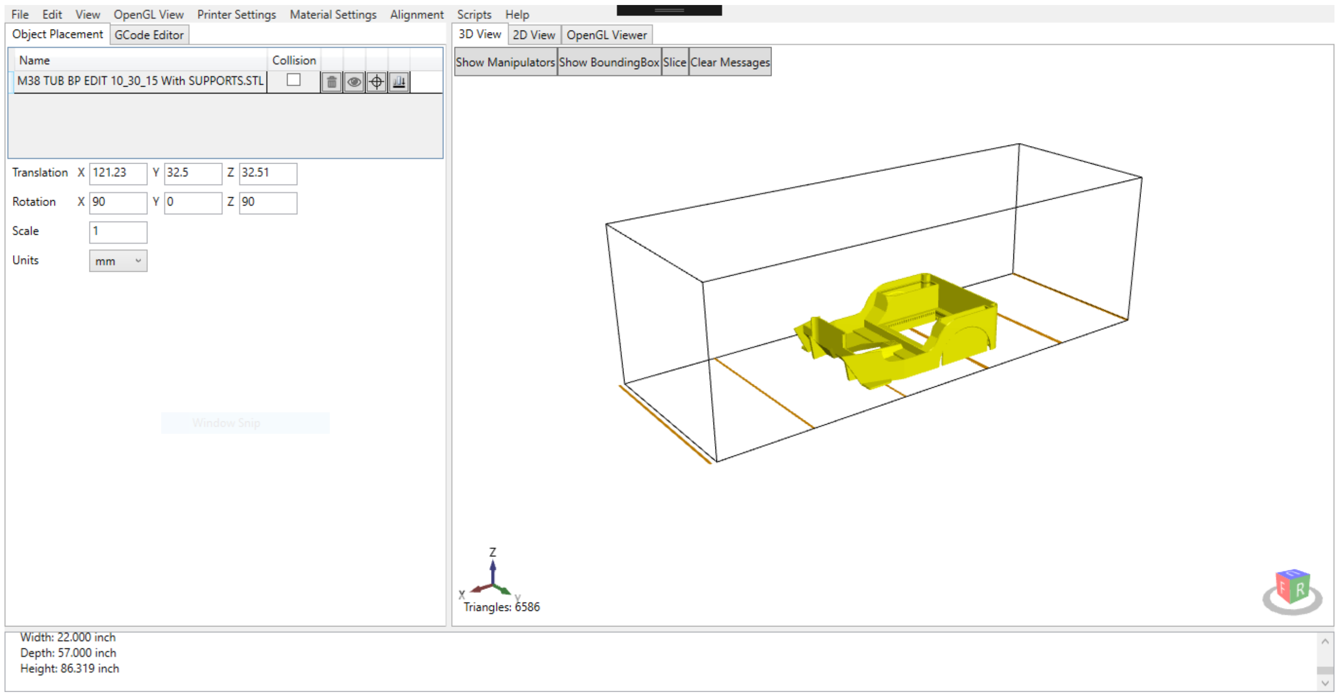
Introduction

Additive Manufacturing (AM), more commonly known as 3D printing, is the process of building a part layer by layer. The process involves adding material to fabricate a part rather than subtracting material as in traditional manufacturing. AM can use many materials such as polymers, metals, and composites [1]. 3D printed parts are designed in computer aided design (CAD) software and then turned into toolpaths, or G-Code, using a program known as a slicer.

Slicing works by loading in a stereolithography (STL) file, which is a representation of a solid model using triangular faces [2]. The file is then cut into “slices” by intersecting a horizontal plane with the STL file multiple times to form the layers. After slicing, each layer is stored as a polygon that represents a layer of the part. The slicing software then fits toolpaths to the polygon based on the user specified settings. This involves creating perimeters, the area on the outermost and innermost area of the part, and infill, the central areas of the part.

ORNL Slicer was created in 2014 when the need arose for a slicing software designed around the capabilities of BAAM (Big Area Additive Manufacturing). BAAM is used for making large parts reaching 20 feet long but has the potential for printing without size boundaries [3,4]. The already existing slicers, such as Slic3r, Cura, and Skeinforge, did not account for the specific needs of BAAM because of the drastic change in printing size. These slicers could not load in large files, use arc moves, or perform advanced extruder commands. Since its creation, the ORNL Slicer has been constantly developed and adapted to meet the needs of BAAM as well as other additive processes. The original slicer was developed as two separate programs, a user interface written in C# (see Figure 1) and a slicing engine written in C++ that communicated via sockets. Separate programs limited slicer functionality and the ability to easily pass information from the user to the engine. These limitations, along with the desire for a more interactive and capable interface, led to the development of ORNL Slicer 2.

In ORNL Slicer 2, the engine and user interface are combined into one C++ program allowing the user to have increased interaction with the slicing process. In addition to combining the programs into one, the new slicer adds new settings, slicing options, and other features to improve the capabilities of ORNL slicer, which enables it to be more flexible for new machines and processes.



M38 TUB BP EDIT 10_30_15 With SUPPORTS.STL

Translation X 121.23 Y 32.5 Z 32.51
Rotation X 90 Y 0 Z 90
Scale 1
Units mm

Triangles: 6586

Width: 22.000 inch
Depth: 57.000 inch
Height: 86.319 inch

Figure 1: The original ORNL Slicer with a Jeep STL file loaded into the software

Framework

Both ORNL Slicers are primary designed for researching new processes to improve additive manufacturing. As research advanced, so did the controls required to operate large-scale 3D printers. For example, controlling the pitch and roll of an extruder for more options during path planning and better overhang support became a requirement to achieve 5-axis printing. Printing layers on non-planar 3D surfaces is also a developing area of AM research that ORNL Slicer 1 was not capable of facilitating. This exposed the limitations of the first ORNL Slicer as required features were not available or easily added.

ORNL Slicer 2 has been redesigned to have a highly extendable framework to advance research while maintaining all capabilities of the first ORNL Slicer. The framework is designed to keep core elements integrated into the software; however, new modules can be easily added to test new path planning techniques, such as working with spline geometry; new machine types, such as a concrete 3D printer; or completely different ways of slicing, such as a non-layered approach. All of this works on a variety of machines that print different materials, using different processes, and with large variations in size.

Part and Layer Settings

One limitation of the first ORNL Slicer had was the lack of ability to apply unique settings to multiple parts being sliced simultaneously. If two parts were to be printed simultaneously with different settings, they had to be sliced separately. Then the two resulting G-Code files needed to be spliced together. In ORNL Slicer 2, as each file is loaded in, it is automatically assigned a color. The file is given a name (the default will be the file name, but a count will be added to the end in the event of duplicate file names) and added to a list. From that list, each individual file can be selected and a separate settings file can be applied to it. During the slicing process, the engine will use the specific settings for each part and each layer as designated by the user. These settings can vary vastly from part to part and layer to layer.

Additionally, the original slicer could not apply different settings to different regions of the same part. If the user wanted to have sparse infill at the bottom, but dense infill near the top, the slicer couldn't apply those settings. The only solution was to slice the part twice, once with each of the desired settings, and then manually combine the files by picking the desired layers from each G-Code file. This was not ideal because the user had to be certain that both G-Code files had the part sliced in the exact same position. If the part had even the slightest rotation or translation, the part would print misaligned and could fail.

Slicer 2 has solved the issue of not being able to use multiple settings within the same part. Slicer 2 now divides the STL file into layers at a height specified by the user. Once the part has been divided, the user will see a list of these layers and can then apply settings to specific layers as needed. The user will select a default profile of settings to apply to all layers that aren't given specific instructions. With this, the user can make a very sparse or hollow infill in the center of the part, then gradually increase the density towards the top of the part to allow for a solid fill, for example. This also allows for increased speeds to be used for layers that don't need high accuracy. The ability to change the infill density throughout the part, making certain areas hollow and other areas solid, is something not available with traditional manufacturing.

In addition to applying settings to specific part layers, the user can also apply mesh-specific settings. A mesh is a solid body, represented with triangular faces, that is contained within an STL file. A single STL file can contain more than one mesh and can be loaded into Slicer 2 as several meshes. Then, the user can divide the meshes apart as desired. If the user wants to delete unneeded meshes, make copies of specific meshes, or move the meshes, this is now possible with Slicer 2. After the user has the meshes needed in the proper location, settings can be applied to each specific mesh. This is an easy way of slicing different regions of a part using different settings. For example, the ability to assign settings to specific meshes and layers will enable machines that use more than one additive process, such as a metal wire fed and metal powder blown system, to fabricate layers using both processes.

Smarter Infill

Circle Packing

Modern slicers currently rely heavily on geometry for all their calculations. This one-size-fits-all approach to infill causes the user to have to account for the physics of a print, such as thermal properties, when adjusting the settings, such as minimum layer times. Ultimately, this means that infill isn't customized to specific loading conditions. In ORNL Slicer 2, in addition to the traditional infill patterns (lines, grid, concentric, honeycomb, etc.) based on geometry, a new process called circle packing has been added for implementing infill.

Circle packing involves filling the infill region with circles and using the centers of these circles as vertices in a graph [5,6]. Using computational analysis, one can find which areas will endure the most stress. This data can be used to weigh the edges of the graph created with the circles. Once all the circles have been placed and all edges have been weighed, an algorithm needs to be used to traverse the pattern. One such algorithm is the Chinese Postman Problem. This algorithm works to find the optimal path through the list of vertices [7]. In addition to stress modeling, thermal modeling and other factors can be used to weigh the edges and determine the optimal infill pattern.

ORNL Slicer 2 will generate the polygon(s) to represent the infill section, then use circle packing to fill the polygon(s). The user defines the method for generating the circles and then the method for traversing the infill. The result is an optimized infill pattern that saves time, weight, and material. Figure 2 shows circle packing as it is applied to an airplane wing model.

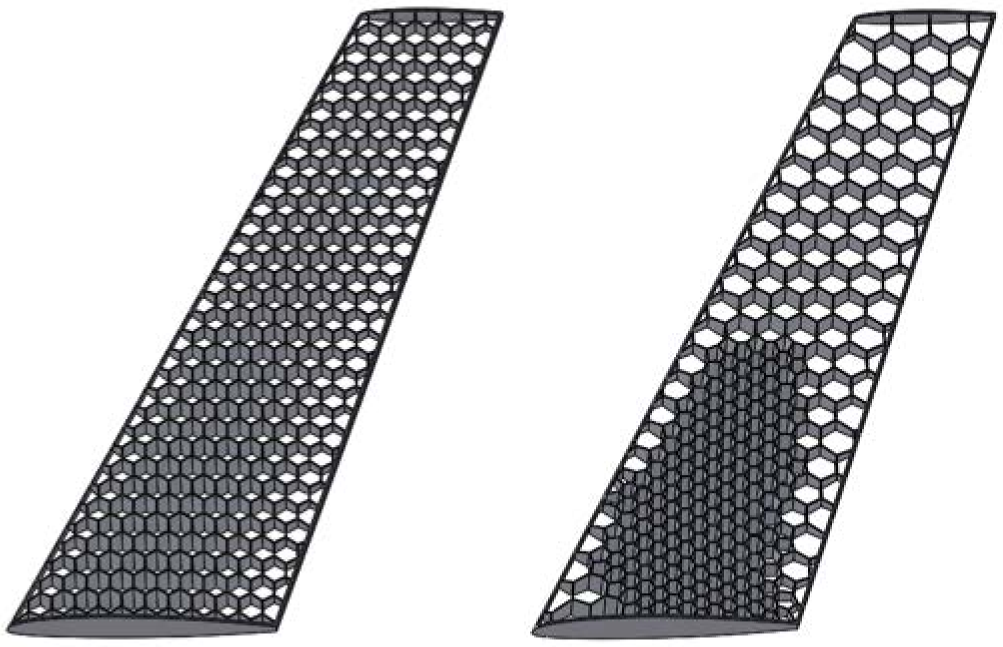


Figure 2: On the left, a wing sample with a uniform geometric infill pattern. On the right, an infill pattern optimized using circle packing

Real-Time Slicing

Feedback

The current workflow for making a 3D printed part is as follows: design the part in CAD, export the file as an STL file, load the STL file into the ORNL Slicer, slice the part, export the resulting G-Code, load the G-Code into the machine, and run the print on the machine. Because the process on the machine is an open-loop (meaning no feedback), if there is an anomaly during the printing process, such as an area that is over or under filled, the part must either be canceled, resliced and reprinted, or the part must be resliced and a new file must be loaded in during the print to try and recover.

These methods rely heavily on user interaction from an experienced operator. ORNL Slicer 2 is being designed with real-time reslicing capabilities. The addition of sensors for monitoring the status and quality of the print will mean real-time closed-loop feedback for each layer during the print. For example, if there is an imperfection on layer 12, the software will see what the defect is and where it occurs so that the slicer can make an adjustment to the tool paths for layer 13. This process will be integrated into the control computer of the BAAM to allow for fully-automated reslicing in real-time. This requires a restructuring of the engine to allow for slicing and reslicing of specific layers that will use collected information for adapting the layer plan. If there is an area that is overfilled, the slicer will need to see that area and isolate it so that more material is not added on top of the buildup. Once the rest of the part is level with the buildup, then the area that was overfilled needs to be re-added into the tool paths so that it gets printed over. This whole process of slicing and reslicing by the host computer must happen in real-time to prevent the machine from having to stop or pause between layers.

Fast Slicing

The original ORNL Slicer, like most other slicers, requires the user to click a button to initiate the slicing process. The slicing engine normally loads in the settings the user had preset and passes the part file via socket to the engine to create the tool paths and G-Code. Each time the user wants to make a small change in the settings, such as increasing the perimeter speed from 10in/s to 11in/s, the entire process must be redone so that the part can be resliced, a new G-Code file created, and the file reparsed. For large BAAM parts, this can be a time-consuming process because it’s not uncommon to have 500,000+ lines of G-Code for a single part. ORNL Slicer 2 has been reengineered to divide the part into regions so that changes to settings don’t require a full reslice. If the infill density gets changed by the user, the infill region is the only region to be resliced. In addition to dividing the part into regions to save time on reslicing, the slicer is fast enough to run in the background at all times. This prevents the user from having to click a slice button and wait for a loading screen. Instead, the slicing will update each time a setting has been changed, and the G-Code will reparse automatically. The significant improvement to slicing speed is necessary for successful real-time slicing as a long slicing update time would affect layer times and ultimately impact printing success.

Conclusion

The ORNL Slicer is a software package that creates toolpaths and G-Code from 3D CAD files (STLs). It was designed specifically for BAAM but has since been extended to many other processes and machines. The original version was limited in functionality because of the separate programs required for the user interface and slicing, which prevented the user from having direct interaction with the engine. In the new version, the slicer has been redesigned to allow for an improved framework that allows for part and layer-specific settings, smart infill, and real-time slicing. ORNL Slicer 2 is a faster and more capable slicer that can be easily scaled and adapted to new machines and processes for additive manufacturing. The future of ORNL Slicer will see constant development as the needs of BAAM and other AM processes change. The framework has been specifically designed to allow for adding new research areas as new ideas and challenges arise, such as 5-axis printing and non-planar printing.

Acknowledgements

This research has been sponsored by the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy at Oak Ridge National Laboratory.

References

1. P. C. Joshi et al., "Direct digital additive manufacturing technologies: Path towards hybrid integration," 2012 Future of Instrumentation International Workshop (FIIW) Proceedings, Gatlinburg, TN, 2012, pp. 1-4.
2. Gao, Wei, et al. "The status, challenges, and future of additive manufacturing in engineering." Computer-Aided Design 69 (2015): 65-89.
3. Holshouser, Chris, et al. "Out of bounds additive manufacturing." Advanced Materials and Processes 171.3 (2013).
4. Love, L.J., *Cincinnati Big Area Additive Manufacturing (BAAM)*. 2015, Oak Ridge National Laboratory.
5. Kim, Seokpum, et al. “An Integrated Design Approach for Infill Patterning of Fused Deposition Modeling and its Application to An Airfoil” Society for the Advancement of Material and Process Engineering (SAMPE) Conference, Seattle May 2017.
6. Dreifus, G., et al. “A new approach to tool path generation in additive manufacturing, in Symposium on Computational Fabrication.” 2017: Cambridge, MA.
7. Dreifus, G.D., et al. “Path Optimization Along Lattices in Additive Manufacturing Using the Chinese Postman Problem.” 3D Printing and Additive Manufacturing, 2017.