Title:
Using Topological Data Analysis to Infer the Quality in Point Cloud-based 3D Printed Objects

Authors:
Paul Rosen, prosen@usf.edu, University of South Florida
Mustafa Hajij, mhajij@usf.edu, University of South Florida
Junyi Tu, junyi@mail.usf.edu, University of South Florida
Tanvirul Arafin, tanvirul@mail.usf.edu, University of South Florida
Les Piegl, lespiegl@usf.edu, University of South Florida

Keywords:
3D Printing, Point Cloud, Topological Data Analysis

DOI: 10.14733/cadconfP.2018.xxx-yyy

Introduction:

3D printing of parts is gaining incredible popularity in manufacturing. However, assessing the quality of models before they are printed remains a challenging problem, particularly when you consider point cloud based models [3], such as those that come from 3D scanners. This paper introduces an approach to quality assessment, which uses techniques from the field of Topological Data Analysis (TDA) to compute a topological abstraction of the eventual printed model. This abstraction enables investigating certain qualities of the model, with respect to print quality, and identify potential anomalies that may appear in the final product.

Mapper and Persistent Homology:

This approach uses 2 of the fundamental tools of TDA, namely Mapper [4] and persistent homology [1], to provide users with feedback about their models (see Figure 1). We use Mapper to extract information about the layer-by-layer connectivity of the model to be printed, providing an abstraction of the overall shape of the object. Persistent homology on the other hand is a tool that normally is used to provide a multiscale view of connected components, holes/tunnels, and voids in data of any dimension. Our approach uses persistent homology for the detection of connected components and holes within a printer layer.

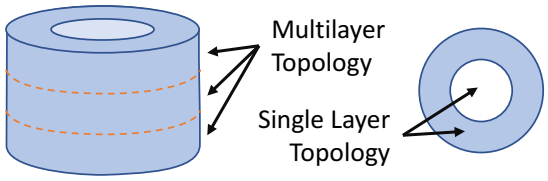

Fig. 1: Our approach uses Mapper to look at the topology of multiple layers (left) and persistent homology to understand the topology of a single layer (right).

The inner workings and associated details of both Mapper and persistent homology are quite complicated, and so we refer the reader to prior work for a better understanding [1, 4]. We will instead provide an intuition about the types of structures captured by each of these tools.

Mapper

Mapper is a TDA tool that provides a graph-based abstraction of the topology of a mesh or point-based data. Mapper construction starts by first slicing the data, vertically in our case. Next, graph vertices are created from connected components identified within each layer. In other words, the connected components of the layer are "collapsed" into a graph vertex. There are many variations on identifying connected components from points. We use the persistent homology approach, introduced in the next subsection. Finally, graph edges are added between components that touch on neighboring layers. This connection is made by adding a small amount of overlap to each layer. The resulting graph can describe the overall topology of the connected components of a printed object.

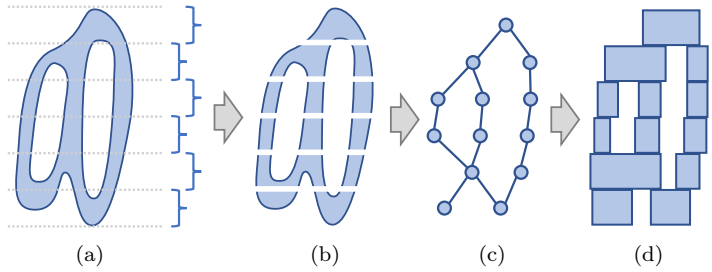

Fig. 2: Example of Mapper on a mesh. The (a) model is (b) sliced. (c) Connected components are collapsed to vertices and edges added for components that touch. (d) Finally, an illustration of the printed object is shown.

Figure 2 shows an example of Mapper on a simple domain. First, (a) the input model is (b) sliced with layer thickness being set to equal the 3D printer's layer resolution. Next, (c) the connected components are found and edges added when they touch. Finally, an illustration of the printed object is shown. As can be seen in this illustration, the nodes of the Mapper graphs do not provide any insight into the size or shape of a given connected component. Instead they provide insight into which components touch and how those components may or may not form holes in the output model.

The calculation of Mapper is relatively inexpensive. The slicing operation is linear in the number of points. The connected component detection is naively quadratic in the number of points in a layer, but this can be improved with spatial partitioning. The overall performance can be improved by using a parallelized and distributed algorithm [2].

Persistent Homology

Given a topological space $\mathbb{X}$, the homology groups $\mathsf{H}_0(\mathbb{X})$, $\mathsf{H}_1(\mathbb{X})$, and $\mathsf{H}_2(\mathbb{X})$, describe the connected components, holes/tunnels, and voids of the space, respectively. For example, consider the torus is Figure 3. It has a single connected component. It has 2 tunnels, the hole through the middle of the torus and the tunnel through the ring of the torus. Finally, it has 1 void, the space inside of the torus.

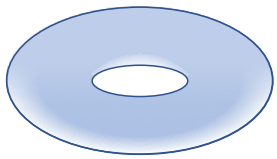

Fig. 3: A 3D torus.

The multiscale notion of homology, called persistent homology, extracts the homology groups of a set of points considering different resolutions. The topological feature therefore has a minimum resolution where it first appears, known as the birth time, and a maximum resolution it is visible, known as its death time. This can be intuitively thought of as the thickening of points. Figure 4 shows an example. Starting with (a) 12 points, the points are thickened, until (b) they form a single connected component with a hole. As the points continue to thicken (c) the hole remains visible, until (d) the thickness of the points closes it.

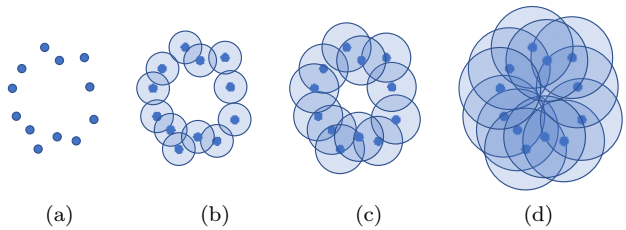

Fig. 4: Example of persistent homology. As points are thickened, from (a) to (d), a hole/tunnel forms in (b) and closes in (d).

Calculating connected components is relatively fast. Naively it is quadratic in the number of points per layer, but optimization leads to fast performance. Finding the H_1 homology groups (i.e. holes/tunnels) in persistent homology is quite expensive. The worst case performance is cubic in the number of points. The average run time is linear with a large time constant. We mitigate this by pre-extracting per-layer connected components and running this calculation only on those components.

Link Between Mapper and Persistent Homology
The most direct link between Mapper and persistent homology is to use persistent homology approach in the calculation of $\mathsf{H}_0(\mathbb{X})$ homology groups (i.e. connected components) for the individual slices of the Mapper algorithm. However, we augment the conventional Mapper implementation by further attaching the $\mathsf{H}_1(\mathbb{X})$ homology groups (i.e. holes/tunnels) to the individual nodes of the Mapper graph. By doing this, the number of holes in each connected component retained for further analysis.

The Topology of 3D Printing:
It turns out that both Mapper and persistent homology have direct applications to the 3D printing problem. For Mapper, the slicing operation has a direct corollary in the layers of a 3D printer. Therefore, the slice thickness, known as the cover, can be set to the same value as the thickness of a single layer on the 3D printer (i.e. the z resolution). For persistent homology, the calculation of connected components is the same as a physically connected components within a single layer. The holes within each layer represent the holes within the model. These can be determined by targeting the xy resolution of 3D printer of interest.

Visualization
Once the topology of the point cloud has been calculated, we provide a visualization for inspecting the data. The visualization contains 3 components. The first, and most important, is the Mapper graph of the topology, as seen in Figure 5 (left). The Mapper graph shows the individual connected components of the model. In addition, each tunnel going through the connected component is represented by a point in the node visualization. The next 2 visualizations are: the 3D point cloud (Figure 5 (upper right)), with regions highlighted based upon the selection of Mapper graph nodes, and a 2D slice visualization (Figure 5 (lower right)), again based upon nodes selected in the Mapper graph.

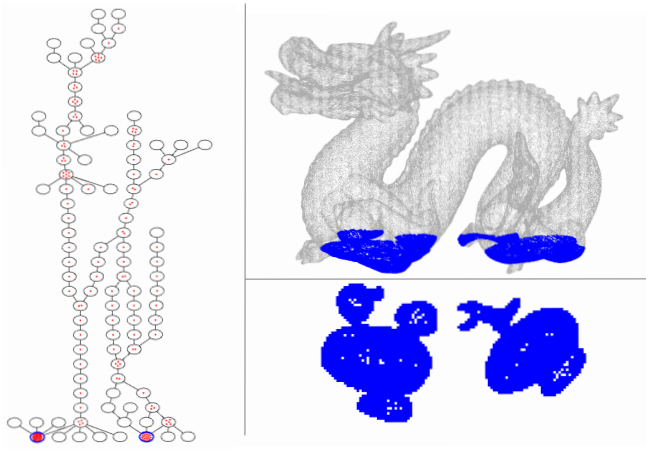

Fig. 5: Our software with the Standford Dragon dataset.

Results:
We implemented our approached using a number of tools. First, we implemented Mapper in Java. Our software would load a point cloud, slice it, detect connected components, and export the Mapper graph and connected component points. Each connected component would then be fed into Ripser[1] for persistent homology detection of holes/tunnels. For the visualization of the Mapper graph, the layout was

[1]Ripser: https://github.com/Ripser/ripser.

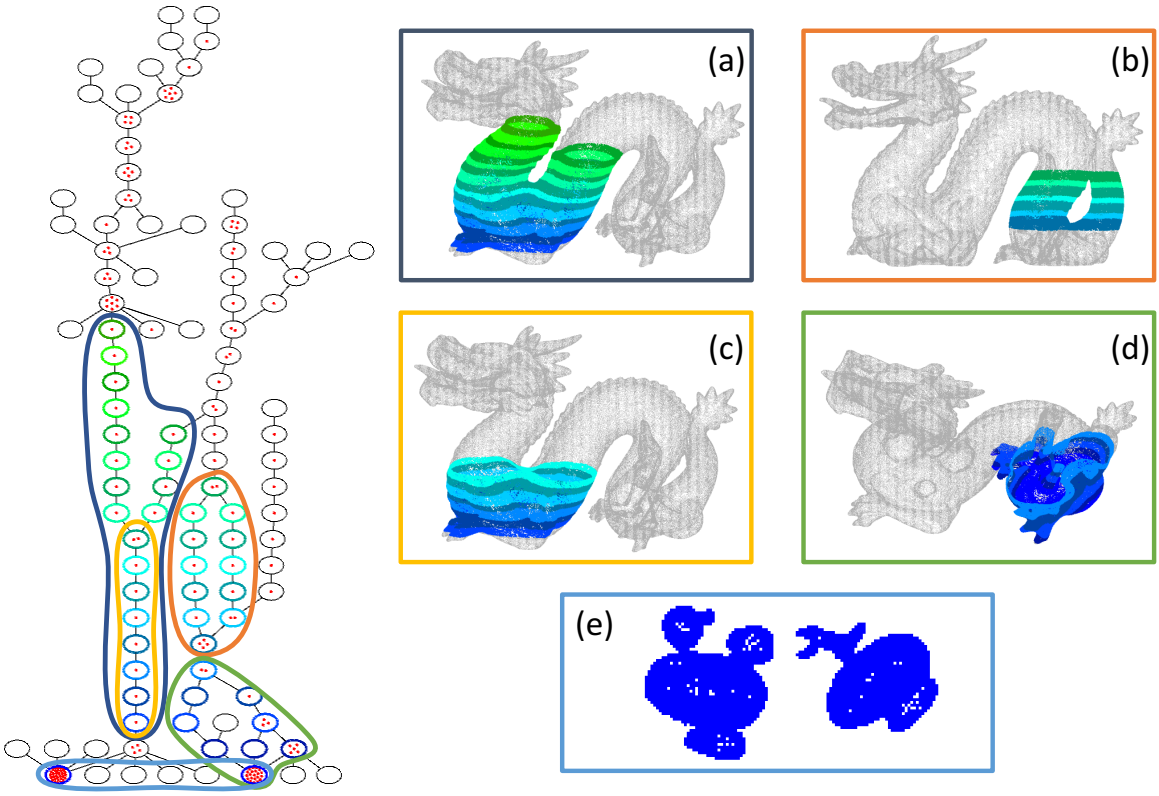

Fig. 6: Results of Dragon dataset.

calculated using Graphviz[2]. The data was then fed into our visualization tool built using Processing[3].

We tested our approach on the Dragon dataset from the Stanford 3D Scanning Repository. We used the points from the reconstructed dataset, which contained approximately 566,098 points. The question we were after was, if someone was to try to rasterize these points directly for 3D printing (ignoring any mesh connectivity), what sort of anomalies would occur. We first scaled the model to a height of 10 cm. We then chose the z resolution to be 3.3 mm and xy resolution to be 1.0 mm.

After running our pipeline, the results are displayed in Figure 6. The tree on the left overviews the entire structure of the graph. We will concentrate on the few circled regions.

First, starting with Figure 6 (c) in yellow, notice that this region represents a portion of the body of dragon. In this region, each ring forms a single connected component, each with a single hole through the middle. That is until the topmost ring, where a single connected component has 2 holes, beginning the bifurcation of the upper front and middle portions of the body, as seen in Figure 6 (a) in dark blue. This feature can be observed in the graph by looking at the top most node in the yellow circle. Notice 2 red dots, indicating 2 holes in that component.

Next, notice the region Figure 6 (b) in orange. In this region, the model itself splits and comes back together leaving a hole between the torso and tail. This can be observed in the graph as well. Starting after the bottom node of orange region, the graph bifurcates, indicating a split in the connected

[2]Graphvis: https://www.graphviz.org/.
[3]Processing: https://processing.org/

components, and merges again at the top. This splitting and merging pattern is indicative of an exterior hole in the model. This same type of splitting and merging behavior can also be noticed in the graph region circled in green and associated with Figure 6 (d). This hole is caused by the leg and body coming together. However, it is difficult to observe by looking at the 3D imagery of the point cloud. In fact, we could not find a good viewing angle that showed this hole directly.

Finally, we look at the bottom slice of the model in Figure 6 (e) in light blue. Looking at the graph, one may observe 2 nodes on the bottom layer that have many points in the visualization. Each point representing a hole in the layer. This may represent a problem for watertightness, particularly given that this is the bottom layer. Observing the connected components represented by those 2 node in Figure 6 (e), many holes are visible in the layer due to inadequate resolution of the points. The initial concern about watertightness is confirmed, given that these holes are not covered by a subsequent layer.

Conclusions:

In conclusion, we have presented an approach for using Topological Data Analysis in the evaluation of point cloud models quality in 3D printing. We made some simplifying assumptions in this paper. For example, we assume that 3D printing resolution is uniform across the entire xy domain, which is not necessarily true. We also chose a naive rasterization procedure, though any other procedure could be chosen. In fact, any pre-rasterized model would be adequate for analysis in this pipeline. It is also important to note that this approach does not necessarily report specific problems, but it instead provides a framework for identifying regions where certain problems may exist. In the future we will expand our study by not only studying the topology of the positive space, but also the topology of the negative space. This will provide new direct insights about features like watertightness and connectivity of the holes within the object.

Acknowledgement:

This work was supported in part by the National Science Foundation (IIS-1513616).

References:

[1] Edelsbrunner, H.; Letscher, D.; Zomorodian, A.: Topological persistence and simplification. In Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on, 454–463. IEEE, 2000. http://dx.doi.org/10.1109/SFCS.2000.892133.

[2] Hajij, M.; Assiri, B.; Rosen, P.: Distributed mapper. arXiv preprint arXiv:1712.03660, 2017.

[3] Oropallo, W.; Piegl, L.A.; Rosen, P.; Rajab, K.: Point cloud slicing for 3-d printing. Computer-Aided Design and Applications, 15(1), 90–97, 2018. http://dx.doi.org/10.1080/16864360.2017.1353732.

[4] Singh, G.; Memoli, F.; Carlsson, G.: Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition. In Eurographics Symposium on Point-Based Graphics, 2007. http://dx.doi.org/10.2312/SPBG/SPBG07/091-100.