NozMod: Nozzle Modification for Efficient FDM 3D Printing

Daphna Kaplan
Technion - Israel Institute of Technology
Haifa, Israel
daphnakaplan@campus.technion.ac.il

Shir Rorberg
Technion - Israel Institute of Technology
Haifa, Israel
shiror@campus.technion.ac.il

Mirela Ben-Chen
Technion - Israel Institute of Technology
Haifa, Israel
mirela@cs.technion.ac.il

Yoav Sterman
Technion - Israel Institute of Technology
Haifa, Israel
sterman.yoav@technion.ac.il

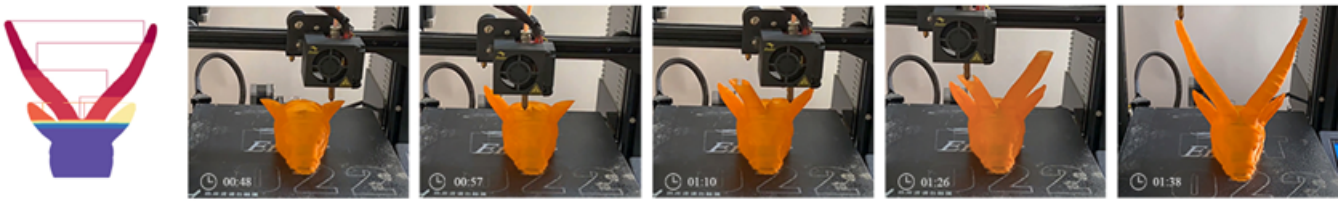


Figure 1: Our nozzle modification allows printing sliced layers out of z-order by grouping them into chunks (left). This leads to less travel movement (during which material is not extracted) of the printer head, as separate "islands" of the model are printed consecutively (center). The final printed model (right) is printed faster and exhibits considerably fewer stringing artefacts compared to layer-by-layer printing.

ABSTRACT

3D printing is based on layered manufacturing, where the layers are printed consecutively in increasing height order. In Fused Depositing Modeling (FDM), the printing head may travel without extruding material between separated "islands" of the sliced layers. These travel movements increase the printing time and reduce the quality of the 3D printed part. We present an extended nozzle modification, which can be applied to off-the-shelf FDM printers, and a corresponding toolpath generation algorithm. Together, these dramatically reduce the amount of travel movement, thus improving the printing time and quality of the results. The extended nozzle allows us to print the sliced layers out of order, where part of a lower layer might be printed after a higher layer was already printed. Our toolpath generation algorithm takes advantage of this capability and generates a toolpath which maximizes the number of consecutive layers printed within the same "island" without requiring any travel movement. In addition, the algorithm optimizes the order of the printed islands, and guarantees that the generated toolpath will not cause collisions between the extended nozzle and the model. We demonstrate our approach on a collection of varied models, and show that we considerably reduce the required travel, and improve the printing time and quality compared to standard layer-by-layer printing.

SCF '22, October 26–28, 2022, Seattle, WA, USA

ACM ISBN 978-1-4503-9872-5/22/10.
https://doi.org/10.1145/3559400.3561999

CCS CONCEPTS

• **Computing methodologies** → **Mesh models**; • **Hardware** → **Emerging technologies**.

KEYWORDS

FDM, toolpath generation, travel movements, nozzle modification

ACM Reference Format:
Daphna Kaplan, Shir Rorberg, Mirela Ben-Chen, and Yoav Sterman. 2022. NozMod: Nozzle Modification for Efficient FDM 3D Printing. In *Symposium on Computational Fabrication (SCF '22), October 26–28, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3559400.3561999

1 INTRODUCTION

3D printing is a manufacturing process in which a 3D model is sliced into layers, and then each layer is added one-on-top-of-the-other until the object is complete. In the Fused Deposition Modeling (FDM) 3D printing process, the 3D printer nozzle moves planarly while extruding a polymer to produce each layer. Only after a layer is complete does the 3D printer move in the Z direction to start the next layer, and so forth. This ensures that there will be no collisions between the printed model and the nozzle since the nozzle is continuously printing the topmost layer. If the geometry of the model

branches or several objects are printed simultaneously, the slicing layers are divided into "islands" (See Figure 2). In each layer, the printer prints the path that belongs to each island and then moves to the next island. The movements between the islands are called Travel Movements, i.e., movements for which the printer's head moves without extruding material. Travel movements negatively affect printing duration when the printed geometry is branched or if several models are printed simultaneously. Moreover, travel movements might generate stringing, mainly when flexible filaments such as Thermoplastic Polyurethane (TPU) are being used. The strings are removed in a manual post-processing step after printing, thus reducing the quality of the printed parts and lengthening the fabrication process.

We present a method for eliminating most of the travel movement in the FDM 3D printing process. This is done by leveraging a special extended nozzle that we designed and manufactured. The nozzle replaces a MK8 standard nozzle which can be used with most low-cost desktop 3D printers. The extended nozzle enables 3D printing of the layers not necessarily in the order given by their distance from the build platform. This means that the printer can print part of a high layer and then move down to print part of a lower layer (See Figure 1). Most of the unnecessary travel can therefore be eliminated by efficiently re-ordering the slicing layers. In our process, we group the sliced layers into chunks. Each chunk can be as high as the length of the nozzle minus a few millimeters for clearance. As a result, travel movements only exist between the chunks. This leads to more efficient printing than the standard workflow, especially for models that include many travel lines. In addition, our work considerably reduces stringing artefacts. Using our process, the 3D printer nozzle is no longer guaranteed to be at the printed model's topmost layer, and therefore collisions might occur between the nozzle and the printed model. We propose a toolpath generation algorithm which computes the chunks and finds a correct printing sequence which avoids such collisions. Moreover, the algorithm automatically adjusts the number of layers in each chunk to ensure printability when joining branched geometry. To summarize, our main contributions are (1) the design of the extended nozzle, and other required hardware modifications for the 3D printer; and (2) the algorithm that generates a collision-free printing sequence, which takes advantage of the modified nozzle for minimizing travel movement and shortening printing time. We demonstrate our approach with a collection of 3D printed objects that show a significant reduction in travel distance and improved printing quality.

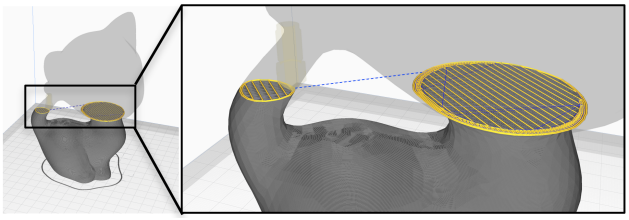


Figure 2: Preview of the printing sequence of the kitten model in Cura [McNeel and Associates 2022]. In this section of the model, each layer is divided into two "islands." The printer's head moves between these "islands" in each printing layer. A travel movement (marked in blue) connects the "islands".

2 RELATED WORK

2.1 Nozzle Adaption In FDM Printing

FDM printer nozzles are typically made of high thermal conductive metal such as brass or copper [Kedare et al. 2020]. The metal qualities and thermal conductivity play a major role in the printed part's surface quality and accuracy. To control the nozzle temperature and make sure the material does not cause leakage or clogging, a heat sink and a fan are installed directly above the nozzle [Kedare et al. 2020; Shaik et al. 2021; Srinivasan et al. 2021]. Therefore, most commercial 3-axis printers' heads are short and wide. Adaptations to the extruder head have been proposed for different needs. For example, when using curved layer fused deposition modeling (CLFDM) the nozzle might collide with the printed part [Nisja et al. 2021]. To address this, in Curvislicer [Etienne et al. 2019] the metal structure which holds the fans is removed from the Ultimaker2 printer to allow more freedom of movement. In Digital Dexterity [Isabella and Tim 2018] a customized nozzle is implemented to avoid collision between the nozzle and the printed material. In Li et al [Li et al. 2020] this issue is addressed by using a 6-axis robotic arm printer that can change its orientation to avoid collisions. In Liu et al [Liu et al. 2021a] a very long nozzle is implemented by moving the heating ring towards the end of the nozzle. However, this nozzle would not fit into a standard printer since it is made for a specific 6-axis printer. In contrast, we propose a nozzle modification which is simple, low-cost, and applicable to many off-the-shelf FDM printers. Combined with our slicing algorithm, our nozzle modification can improve the speed and quality of the resulting printed parts.

2.2 Surface Quality (Stringing)

Stringing artefacts occur due to mid-air traveling within the toolpath, combined with non-optimal printer settings. Bad temperature and wrong retraction settings may cause the filament to leak during travel movement and stick to the model, leading to stretched strings, as reviewed by Shaik et al. [Srinivasan et al. 2021]. In some cases, the stringing phenomena is studied as a desired feature for form exploration during the design process as in Digi-Craft [Lim 2018], or for its hair-like visual effect as in Laput et al. [Laput et al. 2015]. However, in most cases, it is an unwanted defect that requires post-processing of the model [Miller 2021]. One approach to avoid stringing and achieve the best surface quality is by optimizing the printer settings, focusing on the retraction distance and speed [Haque 2020]. However, this optimization must be repeated for every printer and material separately, inevitably requiring many printing tests. In Chiu et al. [Chiu et al. 2016]. a methodology is presented for classifying the printed surface quality and suggesting guidance for the printer settings. Other studies try to oversee the creation of strings: in Paraskevoudis et al. [Paraskevoudis et al. 2020] a convolutional neural network is trained to recognize images with stringing for detection and adjustment. In our work we improve the surface quality and avoid stringing by reducing the travel movement instead of manipulating the printer parameters.

2.3 Toolpath Generation And Slicing

Making high-fidelity parts on a desktop 3D printer is a time-consuming process. A few approaches were proposed to quickly create volumetric models that allow prototyping. In wirePrint [Mueller et al. 2014a] the concept of "low-fi fabrication" is introduced as a method of producing rapid prototypes by generating a wireframe of the model. Another time-efficient method is combining non-printed structures with printed parts for detailed and complex shapes, as done by FaBrickation [Mueller et al. 2014b]. Another option for saving on production time is by printing thin shells. Printing shells, also known as thin walls, is a common approach for generating a sparse toolpath that depicts the outer walls of the 3D model. For example, the "vase mode" In the Cura slicing software [McNeel and Associates 2022] cuts the printing time significantly. However, for a model that has a few separate components, stringing artifacts would be introduced. Some recent work has focused on optimizing the toolpath continuity for specialized printers. Hergel et al. [Hergel et al. 2019] proposed a strictly continuous path to eliminate non-printing travel movements for ceramic printers; Brasoveanu et al. [Brasoveanu et al. 2020] have developed an algorithm to create a thin surface toolpath for ceramic printers that minimizes the number of transitions. Finally, Li et al. [Li et al. 2020] suggested a curved layer toolpath, which reduces travel movements for 3D printing using a 6-axis robotic arm. We, on the other hand , focus on reducing travel movements for printing using off-the-shelf low-cost desktop 3-axis FDM printers. Another approach is optimizing the toolpath of the infill within the model. In this case, toolpath optimization is done within each layer by treating the path as a graph and finding heuristic solutions to the travelling salesman problem, for example in Liu et al. [Liu et al. 2021b], Fok et al uses the ant colony approach [Fok et al. 2019] , and in Aguilar-Duque et al. a genetic simulation is used [Aguilar-Duque et al. 2021]. In Greeff et al. [Greeff and Schilling 2018] the sequence of movements between multiple items is optimized within a single layer. Our work is complementary to these approaches, since we optimize the transitions between different layers.

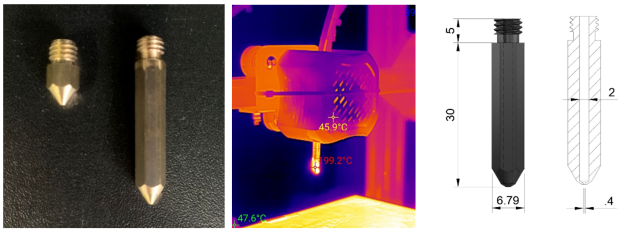


Figure 3: (a) The extended nozzle (right) vs. a standard MK8 nozzle (left). (b) Thermal image of the extended nozzle. (c) Technical drawing of the extended nozzle.

3 HARDWARE MODIFICATIONS

We used an Ender3 desktop 3D printer to 3D print the models. The Ender3, like many other desktop FDM 3D printers, uses a standard MK8 nozzle. The MK8 nozzle has an M6 thread, and its total length is 13mm (including the thread). These nozzles are usually made from brass because this material has good thermal conductivity. In FDM printers, the nozzle is screwed into a heating block that contains a heater and a temperature sensor. Since the heater is located at the heating block at the top of the nozzle, some heat is lost toward the nozzle's tip. When extending the length of the nozzle, the amount of heat-loss increases. To evaluate the amount of heat lose, we used a thermal camera to measure the temperature difference between the temperature at the top of the nozzle and the temperature at the tip. For validation, we compared the temperature at the top to the temperature that the sensor reads and displayed on the printer's screen. The thermal image photos show a decrease of 10 degrees between the top and the bottom of the nozzle, therefore, we recommend increasing the printing temperature by 10 degrees, for example, using 210 degrees Celsius instead of 200 when printing PLA. The design of the extended nozzle is based on the MK8 nozzle design; therefore, the nozzle can be mounted on any 3D printer that used MK8 nozzles. The length of the extended nozzle is 35mm, including 5mm thread. The cross-section is hexagonal for easily replacing the nozzle using a wrench. The hole size of the nozzle is 0.4mm. See Figure 3 for the nozzle design. An additional important modification to the printer is lifting the z-axis limit switch to compensate for the height of the nozzle. Lifting the limit switch prevents the extended nozzle from crashing into the build plate. On the Ender3, the limit switch is mounted on a rail using screws. The limit switch may be lifted and tightened in an upper position (figure 4).

4 TOOLPATH GENERATION

4.1 The Setup

A toolpath is an ordered set of points that the nozzle head follows to extrude the printed material. Our goal is to compute a toolpath for the 3D printer, given an input triangle mesh. We assume that the mesh is oriented with the positive $\hat{z}$ direction pointing "up". As is common in FDM printing, we first slice the 3D mesh into flat layers, where each layer consists of a group of closed polygons. We denote each such closed polygon as a path γ_j^i, where i denotes the layer,

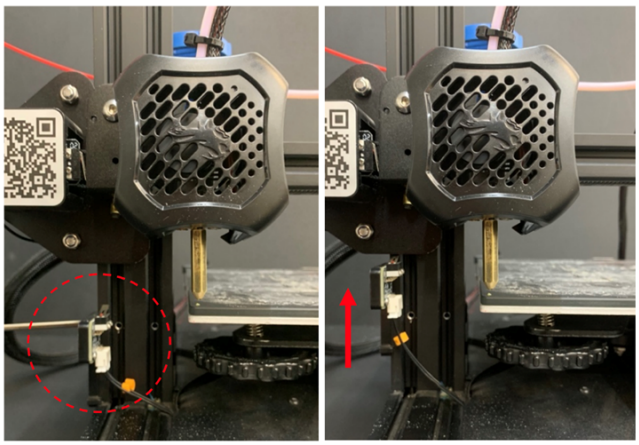


Figure 4: Before (left) and after (right) lifting the z-axis limit switch.

and j is an index of the path within the layer. To compute the paths, we first compute the $\hat{z}$ iso-lines of the mesh, and then extract the connected components within each layer (see figure 5(a)). Our goal is to construct the output toolpath, such that it has the following properties:

- All the points on all the paths γ_j^i belong to the toolpath.
- The toolpath is printable, i.e., (1) when a path is printed, the path immediately under it has already been printed, and (2) the nozzle does not collide with any printed part.
- The toolpath is as short as possible, i.e. travel movement between paths of different "islands" is minimized.

In classic toolpath generation algorithms, the printability constraints are respected by guaranteeing that if a path γ_j^i appears before the path γ_l^k in the toolpath, then necessarily, $i \leq k$. This means that all the paths of the i-th layer must be printed before any component of the i+1-th layer is printed. This is necessary, since otherwise the printer nozzle may collide with parts that have already been printed, or print in mid-air. However, this leads to extensive back and forth travel between different "islands" – paths on consecutive layers which are "on top of each other".

Our extended nozzle allows us to print the paths out of order, such that a path that belongs to a higher layer can be printed before a path from a lower layer, if the vertical distance between them is smaller than the height of the extended nozzle. To facilitate this, we divide the layers into multi-layers, denoted by S_k, which include a collection of layers at consecutive heights, whose maximal vertical distance is given by the nozzle height h_m (see figure 5(b)). This allows us to construct chunks, i.e., ordered sets of paths which are printed sequentially without any travel, thus reducing unnecessary travel movement. We propose an algorithm that generates the toolpath from the input paths γ_j^i. The full problem, where all the points on all the paths can be visited in any order that does not violate printability, is an instance of the Sequential Ordering Problem [Escudero 1988], also known as the Traveling Salesman Problem with Precedence Constraints [Ahmed and Pandit 2001].

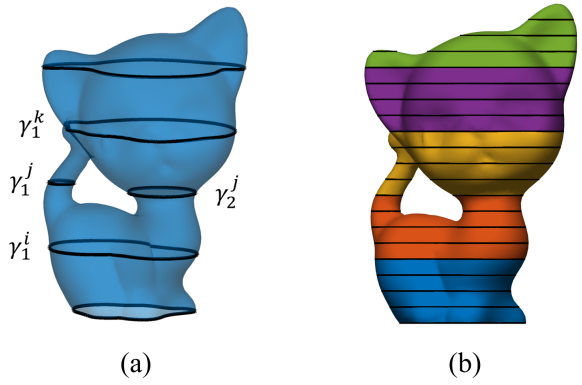


Figure 5: (a) Some iso-z height layers and paths. A path γ_j^i is the j-th connected component of the i-th layer. (b) Multi-layers S_k: a collection of layers at consecutive heights, whose maximal vertical distance is bounded by the height of the extended nozzle h_m.

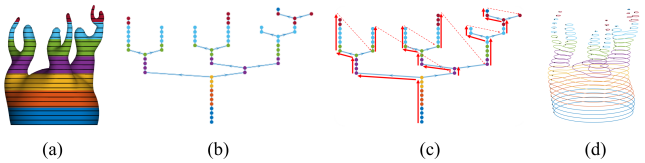


Figure 6: (a) A model and its paths (black curves), colored by its multi-layers S_k. (b) The corresponding z-dependence path graph G. Each node in the graph is a path – a connected component of a height layer. (c) The traversal order defined by the DFS tree T of the path graph G in figure 6(b). (d) The corresponding starting points p_j^i (black points).

This problem is known to be NP-hard, and we propose a simple heuristic using shortest Hamiltonian paths and topological sort. We demonstrate that even this simple approach considerably reduces travel distance since we can print the layers out of order due to the extended nozzle. We believe that additional optimizations are possible and leave further explorations to future work. The algorithm has three parts, that handle z-dependence (Sec 4.2), travel movement optimization (Sec 4.3) and nozzle collision-avoidance (Sec 4.4),respectively.

4.2 Initial ordering by z-dependence

First, we require a notion of "path z-dependence", to guarantee that when a path is printed, the path immediately under it has already been printed.

Definition 4.1. A path γ_j^{i+1} z-depends on the path γ_j^i if and only if the XY bounding boxes of γ_j^i, γ_j^{i+1} overlap.

The z-dependence path graph is a directed graph, denoted by $G = (V, E)$. Its nodes, V, are all the paths in all the layers γ_j^i and for all i we have that $(\gamma_j^i, \gamma_k^{i+1}) \in E$ if and only if γ_k^{i+1} z-depends on γ_j^i. Figure 6 shows an example of a model, its paths, multi-layers and corresponding path graph. Note, that in general, this graph is not necessarily connected, e.g., if the input model has multiple components.

Any toolpath must contain all the paths, and each path can appear only once in the toolpath. In principle, any order that includes all the points on all the paths and is printable is valid. However, optimizing over all possible toolpaths in this setting is computationally prohibitive, and thus we assume that each path is printed consecutively. Hence, a toolpath is uniquely represented as a traversal of a spanning tree (or spanning forest) of G, up to the choice of (1) a starting point, and (2) a clockwise or counterclockwise travel direction for each closed path. A consistent orientation leads to smoother printer head movement, and thus we orient all paths consistently counterclockwise. We build the traversal in a few steps. First, we use an initial depth-first traversal of the path graph by running a depth first search (DFS) on G. We denote the resulting DFS tree (resp. forest, see the second half of section 4.6) by T (see Figure 6(c)). The initial tree T provides the path order, which we use to compute the starting point of each path. Later, we will reorder T by considering the Euclidean distance between the starting points and collision avoidance.

4.2.1 Path Starting Points. Let $\{\gamma_1, \gamma_2, ..., \gamma_N\}$ be the graph traversal of G induced by the DFS tree T, and denote the starting point of a path γ by $s(\gamma)$. For the first path in the traversal, γ_1, we choose an arbitrary starting point. The starting point $s(\gamma_{k+1})$ is chosen as the point on γ_{k+1} which is closest to $s(\gamma_k)$ (figure 6(d)). The starting point of the path γ_j^i is denoted by p_j^i. This choice of starting points generates a smooth "seam".

4.3 Travel movement optimization

4.3.1 Optimizing travel movement within a single layer. When the model branches, a path γ_j^i will have multiple descendants in the DFS tree T. Thus, there are multiple possible traversal orders, depending on the order that these descendants are visited. For example, in Figure 6(c) we can first visit either the left subtree or the right subtree, at each branching point. To minimize the travel movement, we re-order T, such that the successors $\{\gamma_{k_1}^{i+1}, ..., \gamma_{k_l}^{i+1}\}$ of each node γ_j^i in T, are ordered by the shortest Hamiltonian path of their starting points $p_{k_1}^{i+1}, ..., p_{k_l}^{i+1}$. We denote the ordered tree by T^O. To propagate the ordering to the descendants in the tree, and thus re-order all the subtree and not just the immediate descendants, we run DFS on T^O. The result is a DFS order in which the subtrees are ordered by the shortest Hamiltonian path of their starting points (figure 7).

4.3.2 Optimizing travel movement within a multi-layer. The extended nozzle allows us to print layers out of order, if their height difference is smaller than the nozzle height, namely if they are in the same multi-layer. Therefore, the final print order is composed of sequential ordering of the print order of each multi-layer s_k. To obtain this order, we consider the forest$F_k = T^O[S_k]$, which is the sub-graph of T_O induced by the paths which belong to s_k (see Figure 8). To further minimize the travel distance, we reorder the trees of F_k (i.e., the weakly connected components of F_k), by computing a Hamiltonian path on the starting points of the lowest path in each tree. We denote the final order obtained for s_k by $\pi_k = \gamma_1, ..., \gamma_m$ with $\gamma_l \in V$.

4.4 Avoiding nozzle collisions

Nozzle collisions may occur within the same multi-layer, if one path obstructs the access of the printing nozzle to a different path (see figure 9). We therefore generalize the definition of z-dependence to include potential nozzle obstruction. Note that such an obstruction can occur between non-consecutive layers, and also in the same layer, if two paths are closer than the radius of the nozzle.

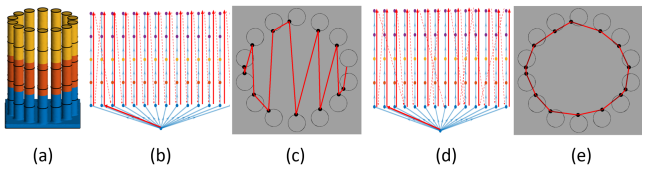


Figure 7: (a) The model and its height layers. (b) The traversal order defined by the DFS tree T. (c) The travel movement resulting from (c). (d) The reordered tree T^O, computed using a Hamiltonian path on the branches at one of the middle layers. (e) The travel movement resulting from (d).

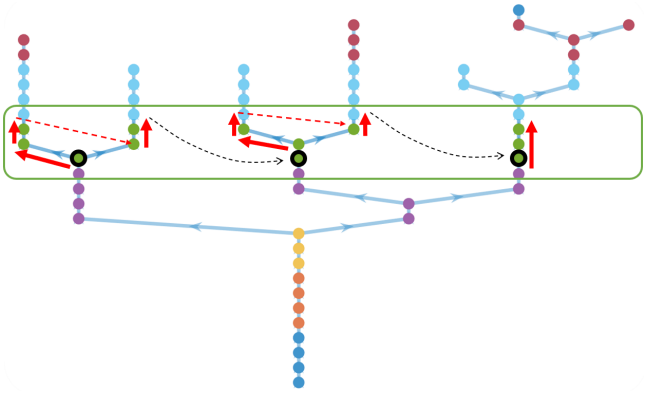


Figure 8: The forest F_k induced by the paths of the multi-layer $s(\gamma_k)$ (green vertices), with the printing order in each tree (red arrows). The vertex representing the first path of each tree is marked with a black outline. The order between the trees (black arrows) is given by a Hamiltonian path on the black vertices.

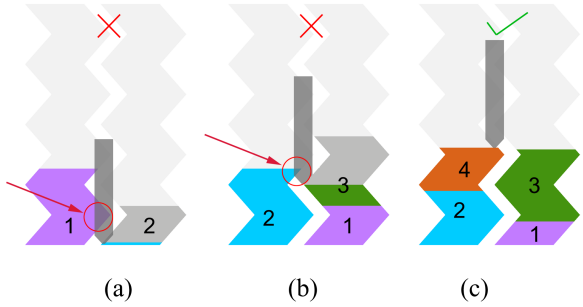


Figure 9: An invalid printing order can cause the nozzle to collide with the partially printed model. (a,b) Non printable orders: the nozzle cannot access and print the grey chunk, since it will collide with the partially printed model (red circle). (c) The order is printable.

4.4.1 The print-dependence graph.

Definition 4.2. A path γ_j^i hides the path γ_m^l if $i \leq l$ and there is an overlap between their XY bounding boxes offset by the half the nozzle radius r/2 (see figure 10(a)).

Similarly to the z-dependence graph G, we construct a print-dependence graph H_k for the multi-layer S_k. Its vertices are all the paths in S_k, and an edge (γ_j^i, γ_m^l) exists if and only if pathγ_m^l hides path γ_j^i (figure 10(c)). This graph may contain cycles of paths in the same layer, as two paths in the same layer might hide each other. For simplicity, we will first assume that H_k does not have cycles, and then explain how cycles are handled.

4.4.2 Print order from a directed acyclic graph (DAG). If H_k is a directed acyclic graph, then we obtain the printing order of its paths using a topological sort algorithm. We construct a set of path-to-print candidates Γ_k which have no incoming edges in H_k. Then, we repeatedly add a path from Γ_k to the printing order, remove it

from the print-dependency graph H_k, and update Γ_k accordingly (see Figure 11, Note that the figure is intended to visualize the algorithm's behavior, and thus the starting point is specifically engineered to generate a case where hiding changes the printing order). Since we only add to the printing order paths which do not depend on any other path (since they do not have incoming edges in H_k), it is guaranteed that the generated toolpath is printable. If there are multiple paths in Γ_k, then in terms of printability they can all be selected to be printed next. We prioritize printing by the order Π_k, computed using Hamiltonian paths in Section 4.3, to reduce travel movement. Specifically, after adding a path γ to the printing order, we add its consecutive path in the printing order Π_k if possible (i.e., if it belongs to Γ_k). If not, we choose the path from Γ_k which has the earliest printing order in Π_k. The final printing order contains all the paths $\{\gamma_j^i\}$. We associate with each path a chunk number, which is incremented when two consecutive paths in the printing order do not belong to the same branch in the tree T.

4.4.3 Handling cycles. Cycles in the print-dependency graph H_k indicate that some paths mutually hide each other. In such cases, the paths must be printed layer by layer, and no chunk optimization is possible. To incorporate this case in the general print order we compute, we first compute the strongly connected component graph of H_k, and denote it by $\hat{H}_k$. This graph is a directed acyclic graph (DAG), where the cycles of H_k become meta-vertices. Thus, we apply the print order computation algorithm for DAGs from the second half of section 4.6. Once a meta-vertex is chosen for addition to the print order, we expand it back to the path cycle, and add all the paths in the cycle consecutively to the print order. Again, we optimize the travel movement by starting the cycle with the path which is closest to last path printed.

4.5 Generalized Dependence

The model may be composed of concentric parts, see for example the model in figure 12(c) which is composed of two nested bowling pins. In this case, with the definitions 4.1 and definition 4.2 for z-dependence and print-dependence, respectively, the algorithm will print the nested parts using a layer by layer order (Figure 12(d)). However, if the distance between the paths of the two parts is bigger than the nozzle radius (Figure 12(a)), then the parts can be printed

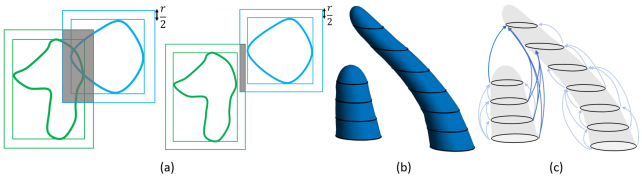


Figure 10: (a) Example of overlapping paths for a nozzle of radius r, the overlap area is given in black. (b) The model and its layers. (c) The print-dependence graph H_1 for the (single) multi-layer S_1. Note that the right part of the model hides the left part, which leads to dependencies in H_k between the left and right parts (dark blue arrows).

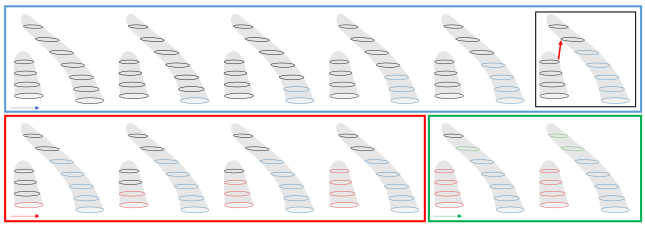


Figure 11: Generating the printing order of the paths from the print-dependence graph H_k. First the paths on the right side of the model are printed, starting from the bottom (blue) until we reach a dependency in H_k between the left and right sides (red arrow). Then we switch to printing the left part (red) until it is done, and then we finish the right part which has no dependencies remaining (green).

independently one after the other, and therefore save the travel movements of printing them layer by layer.

To handle such cases, we replace the z-dependence and print-dependence conditions from definitions 12 and 4.2. In the new conditions for z-dependence and print-dependence, two paths overlap if their respective polygons offsetted by the layer width and r/2 respectively, overlap (Figure 12(b)). The new condition is more computational heavy. One option for decreasing the computation time is to down-sample the polygons for the overlapping check. Figure 12(d,e) shows an example of the resulting chunks using the new condition.

4.6 Implementation details

We implemented our algorithm in Matlab, using its built-in implementation of the traveling salesman problem. Since the number of paths is usually not very high, the computation is fast and for a typical model takes between 10 and 15 seconds on a standard laptop. The algorithm outputs an ordered list of paths and their corresponding chunk number. Then the coordinates of each point in each path are translated into a G-code line, and the extrusion amount is calculated by measuring the distance between every

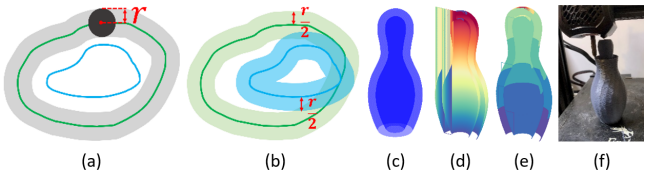


Figure 12: Generalized dependency. (a) The two paths are print-dependent according to condition in definition 4.2 and not print-dependent according to the new condition, (b) the two paths are print-dependent according to the new condition (c) an example of a model with nested parts. (d) The travel lines for a layer by layer printing, (e) chunks and travel lines with the new condition for z-dependence and print-dependence, (f) a frame from the printing process of the model.

two consecutive points. Other settings, such as printing speed and temperature, are added to generate the complete G-code file.

4.6.1 Avoiding collisions during travel movement. While we guarantee that the path order is printable, the travel movements of the nozzle might intersect with the partially printed model. To avoid that, the travel movement is parallel to the XY plane, done at the maximal current printed height. While this simple approach was sufficient, and still led to considerable improvement in travel distance, additional optimizations are possible, e.g. by explicitly computing the intersection between the travel line and the already printed model.

4.6.2 Models with multiple connected components. A model with multiple components (such as the one in figure 10(b)), generates a z-dependency graph G which is a forest instead of a tree. In this case, we add an extra virtual vertex to G, which is connected to the roots of all the trees in the forest. The rest of the algorithm proceeds as usual.

5 LIMITATIONS

The added value of our approach depends on the distance of the travel movement between "islands" compared to the dimensions of the model. For example, models with a single "island", that lead to a z dependency graph which is a directed path (e.g. the Moai model in figure 13), do not lead to considerable improvement compared to layer-by-layer printing. Similarly, models whose dimensions are small compared to the nozzle width will also not benefit from our approach. Due to the hardware adaptations (Section 3) that require altering the position of the printing plate in the z-axis, the maximum height of the printable model is reduced. In our case, the maximum height of models printed by an Ender3 desktop printer is 250mm, however we extended the nozzle by 22mm therefore the maximum height of printable objects is now 228mm. There is a tradeoff between the nozzle height and the printer height. Extending the nozzle enables higher multi-layer groups, leading to better optimization of the printing toolpath but limits the maximal model height. An additional extension of the nozzle would require further research into thermal conductivity to longer distances.

6 RESULTS

6.1 Setup

We slice the models using a layer height of 0.2mm, and defined the multi-layer to be the size of up to 130 layers (max height of 26mm). The printing speed settings are Cura's recommended settings for printing with PLA [Paraskevoudis et al. 2020]: 150mm/sec for travel, 45mm/sec for retraction and 20-30mm/sec for printing. Other settings are also the commercially recommended settings: retraction distance of 5mm, bed temperature of 60°, and 210° for the nozzle [Horvath 2014], 10 degrees more than the recommended temperature for PLA. We used an Ender3 printer for our tests with a 1.75mm PLA filament. Since our model optimizes the order of the printed paths rather than optimizing the infill printing, we demonstrate our results by printing thin-walled models, however printing models with infill is also possible (Figure 13).

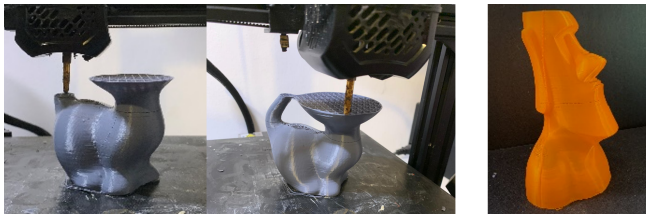


Figure 13: (left) Printing the kitten with infill using our approach. (right) A model for which our method does not provide an advantage since it has a single branch.

6.2 Printing efficiency

We compare our approach to the standard layer-by-layer printing approach to illustrate the effects of our algorithm. The resulting print time (in minutes) and total travel movement distances (in meters) is provided in Table 1 for all the models in the paper. The effect of our method is most pronounced for branched models, such as the Coral in Figure 14. The model's bounding box diagonal is about 29cm, and the total travel distance when printing with the standard layer-by-layer method is 204 meters. When using our method, the total travel distance is only 18 meters, a reduction of 91%. The printing time is also shortened by about 40 minutes. Finally, the printing quality significantly improved, as there are almost no stringing artefacts between the branches. Another use-case of our algorithm is printing multiple models simultaneously on the same printer bed. This is desirable since it enables unsupervised printing of multiple models without manual intervention, while additionally saving time by avoiding the printer setup overhead when changing models. Printing multiple models is possible using the standard layer-by-layer approach, however, it generates multiple stringing artefacts, and therefore it is not commonly employed. Using our method, we get the advantages of multiple model printing, without the disadvantage of the stringing artefacts.

To demonstrate this, we 3D printed a bowling set of 6 pins with bounding box diagonal of about 29cm. Printing using a layer-by-layer approach resulted in a travel movement distance of 251 meters,

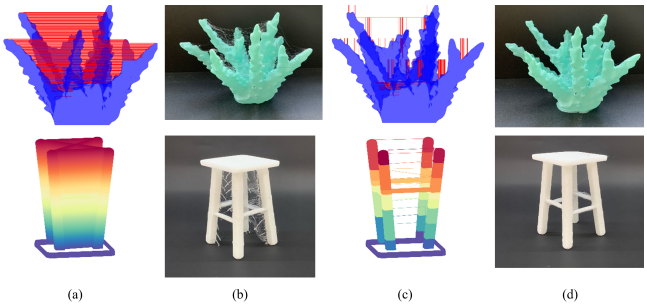
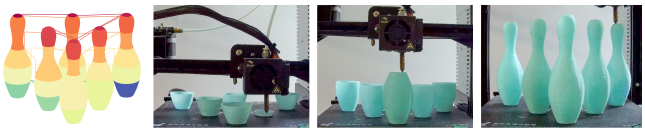


Figure 14: Comparison of standard layer-by-layer printing (a,b) to our approach (c,d). We show the travel movements as red lines (a,c), and the final printed models (b,d). Note the significantly larger amount of travel movement required by layer-by-layer printing (a) and the resulting stringing artefacts (b). See 1 for statistics.

Table 1: A comparison of printing time (without start-up time) (in minutes) and travel movement (in meters) between our approach and layer-by-layer printing.

Models	Print time (in minutes)		Travel movements (in meters)	
	Our algorithm	Layer-by-layer	Our algorithm	Layer-by-layer
Coral	223	262	18.1	204.36
Swan	136	142	9.7	33.2
Yakul	98	114	2.7	87.0
Bowling	326	378	3.2	251.4
ZigZag	59	67	0.4	24.3
Poles	87	102	0.3	41.9
Kitten	113	115	1.3	9.2
Labrinth	71	81	0.3	39.0
Rabbit	111	117	0.3	27.4
Stool	450	596	2544	4811



(a) (b) (c) (d)

Figure 15: Printing multiple models simultaneously on the same printer bed. (a) The chunks (ordered sets of consecutive paths printed without travel) and travel lines generated by our method. (b,c) Frames from the printing process. (d) The final result. See table 1 for statistics.

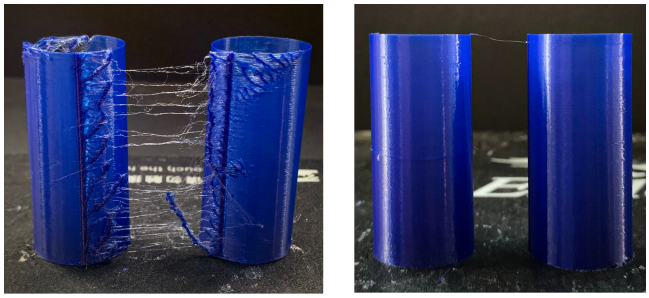
whereas using our approach the travel distance was only 3 meters, a reduction of 99%. The printing time using our method is shortened by 50 minutes. Figure 15. shows the results for this model.

6.3 Surface quality

Stringing artifacts might be generated along travel movements when the printer settings (e.g. retraction speed and retraction amount) are not calibrated, causing pressure in the nozzle during travel. The strings are then manually removed in a post-processing step. Flexible materials such as Thermoplastic Polyurethane (TPU) are especially susceptible to stringing due to their viscosity. Furthermore, TPU strings are more difficult to remove. We considerably reduce the generation of such artefacts by reducing the amount of travel movement. For a simple model of two poles (radius 12.5mm and height 55mm), our algorithm leads to only 3 traveling movements, whereas the layer-by-layer method requires 541 such transitions, leading to considerably more stringing. The visual difference of the model's quality and smoothness can be seen in Figure 16.

6.4 Gallery

Figure 17 shows a gallery of the results of our algorithm applied to different models. The statistics for all the models appear in Table 1. For each model, we show the chunks we computed (each path is color coded according to the chunk it belongs to), the travel movement paths, and the final printed model.



(a) (b)

Figure 16: The surface quality of the same model printed with TPU, using the same printer settings, using the Cura slicer (a) and our algorithm (b). Note the considerable stringing artefacts in (a).

7 CONCLUSIONS AND FUTURE WORK

3D printing, and especially FDM, is known to be a lengthy process. As opposed to classical toolpath optimization methods, which focus on optimizing the toolpath in each layer, our approach focuses on optimizing the layer sequence. This optimization is enabled by a modified, extended nozzle coupled with an algorithm that analyzes the morphology of the 3D model and plans the toolpath while ensuring printability and collisions-free travel between the model and the extended nozzle. We demonstrate our approach on a series of 3D printed models and show improvements in printing time and printing quality, especially when printing with flexible filaments. We believe that our optimization method can be combined with other optimization methods as a complementary solution. This will lead to an even better and more efficient toolpath. Additionally, we plan to investigate opportunities for optimization and creative expression enabled by the extended nozzle, such as freeform and non-planar 3D printing.

Figure 17: A collection of models printed using our extended nozzle and chunks-based toolpath. We show the final printed models and the chunks and travel lines. Note the small amount of travel lines required, which depend only on the branching geometry of the model. See statistics in 1.

ACKNOWLEDGMENTS

The authors acknowledge the support of the Israel Science Foundation (grant No. 1073/21),and the European Research Council (ERC starting grant no. 714776 OPREP).

REFERENCES

Julián I. Aguilar-Duque, Cesar O. Balderrama-Armendáriz, Cesar A. Puente-Montejano, Arturo S. Ontiveros-Zepeda, Jorge L. García-Alcaraz, and Cesar Balderrama@uacj Mx. 2021. Genetic algorithm for the reduction printing time and dimensional precision improvement on 3D components printed by Fused Filament Fabrication. *The International Journal of Advanced Manufacturing Technology* 115 (2021), 2021. https://doi.org/10.1007/s00170-021-07314-w/Published

Zakir Hussain Ahmed and S. N. Narahari Pandit. 2001. . The Travelling Salesman Problem with Precedence Constraints.

Adrian Brasoveanu, Megan Moodie, and Rakshit Agrawal. 2020. As-Continuous-As-Possible Ceramics Printing for Shell Models. *In CEUR Workshop Proceedings, CEUR-WS* (2020), 1–9. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Yu Kai Chiu, Hao Yu Chang, Wan Ling Yang, Yu Hsuan Huang, and Ming Ouhyoung. 2016. A novel real time monitor system of 3D printing layers for better slicing parameter setting. In *UIST 2016 Adjunct - Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. Association for Computing Machinery, Inc, 209–210.

L. F. Escudero. 1988. . Theory and Methodology An inexact algorithm for the sequential ordering problem.

Jimmy Etienne, Nicolas Ray, Daniele Panozzo, Samuel Hornus, Charlie C. L. Wang, Jonàs Martínez, Sara McMains, Marc Alexa, Brian Wyvill, and Sylvain Lefebvre. 2019. Curvislicer: Slightly curved slicing for 3-axis printers. *ACM Transactions on Graphics* 38, 4 (July 2019), 2019. https://doi.org/10.1145/3306346.3323022

Kai Yin Fok, Chi Tsun Cheng, and Nuwan Ganganath. 2019. Herbert Ho Ching Iu, and Chi K. *Tse.* 15, 4 (April 2019), 2277–2287. https://doi.org/10.1109/TII.2018.2889740

Gabriel Pieter Greeff and Meinhard Schilling. 2018. Single print optimisation of fused filament fabrication parameters. *International Journal of Advanced Manufacturing Technology* 99, 1 (October 2018), 845–858. https://doi.org/10.1007/s00170-018-2518-4

Md. Sabit Shahriar Haque. 2020. Minimizing Stringing Issues In FDM Printing. https://doi.org/10.13140/RG.2.2.35536.74247

Jean Hergel, Kevin Hinz, Sylvain Lefebvre, and Bernhard Thomaszewski. 2019. Extrusion-based ceramics printing with strictly-continuous deposition. *ACM Transactions on Graphics* 38, 6 (November 2019), 2019. https://doi.org/10.1145/3355089.3356509

Joan Horvath. 2014. *Mastering* 3 (2014).

Molloy Isabella and Miller Tim. 2018. *Digital Dexterity - Freeform* 3 (2018).

Prashant Kaduba Kedare, S. A. Khan, and Harish Kumar. 2020. 3D Printer Nozzle Design and Its Parameters: A Systematic Review. In *Smart Innovation, Systems and Technologies*. 777–785. https://doi.org/10.1007/978-981-15-2647-3_73

Gierad Laput, Xiang 'Anthony' Chen, and Chris Harrison. 2015. 3D Printed Hair: Fused Deposition Modeling of Soft Strands. In *Fibers and Bristles*, In Uist (Ed.). - Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology, Association for Computing Machinery, Inc, 167–173.

Yamin Li, Dong He, Xiangyu Wang, and Kai Tang. 2020. Geodesic Distance Field-based Curved Layer Volume Decomposition for Multi-Axis Support-free Printing. https://doi.org/10.48550/ARXIV.2003.05938

Chor-Kheng Lim. 2018. *Digi-Craft: A Creative Process in Form-Finding Beyond the Accuracy of* 3 (2018), 258–262. https://doi.org/10.1007/978-3-319-92270-6_36

Hao Liu, Rui Liu, Zhoupeng Liu, and Shuhao Xu. 2021b. Minimizing the Number of Transitions of 3D Printing Nozzles Using a Traveling-Salesman-Problem Optimization Model. *International Journal of Precision Engineering and Manufacturing* 22, 9 (September 2021), 2021. https://doi.org/10.1007/s12541-021-00512-2

Hao Liu, Zhoupeng Liu, and Siting Hao. 2021a. Design of a Throat-extended FDM Extruder for Multi-axis 3D Printing. *Strojniški vestnik – Journal of Mechanical Engineering (April* 2021 (2021), 167–179. https://doi.org/10.5545/sv-jme.2021.7124

Robert McNeel and Associates. 2022. Cura. https://ultimaker.com

Brandon Miller. 2021. How to clean up stringing on a 3D print. https://printingit3d.com/how-to-clean-up-stringing-on-a-3d-print/

Stefanie Mueller, Sangha Im, Serafima Gurevich, Alexander Teibrich, Lisa Pfisterer, François Guimbretière, and Patrick Baudisch. 2014a. WirePrint: 3D printed previews for fast prototyping. In *UIST 2014 - Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, 273–280.

Stefanie Mueller, Tobias Mohr, Kerstin Guenther, Johannes Frohnhofen, and Patrick Baudisch. 2014b. FaBrickation: Fast 3D printing of functional objects by integrating construction kit building blocks. In *Conference on Human Factors in Computing Systems - Proceedings*. Association for Computing Machinery, 3827–3834.

Georg Aarnes Nisja, Anni Cao, and Chao Gao. 2021. Short review of nonplanar fused deposition modeling printing. *Material Design and Processing Communications* 3, 4 (August 2021), 2021. https://doi.org/10.1002/mdp2.221

Konstantinos Paraskevoudis, Panagiotis Karayannis, and Elias P. Koumoulos. 2020. Real-time 3d printing remote defect detection (Stringing) with computer vision and artificial intelligence. *Processes* 8, 11 (November 2020), 2020. https://doi.org/10.3390/pr8111464

Yousuf Pasha Shaik, Jens Schuster, and Aarif Shaik. 2021. A Scientific Review on Various Pellet Extruders Used in 3D Printing FDM Processes. *OALib 0* 8, 2021 (2021), 1–19. https://doi.org/10.4236/oalib.1107698

D. Srinivasan, M. Meignanamoorthy, M. Ravichandran, and V. Mohanavel. 2021. S. v. *Alagarsamy, C. Chanakyan, S. Sakthivelu, Alagar Karthick, T. Ram Prabhu, and S. Rajkumar.* 2021 (2021), 3. https://doi.org/10.1155/2021/5756563