The following publication K. Y. Fok, C. T. Cheng and C. K. Tse, "A refinement process for nozzle path planning in 3D printing," 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, USA, 2017, pp. 1-4 is available at http://dx.doi.org/10.1109/ISCAS.2017.8050471

A Refinement Process for Nozzle Path Planning in 3D Printing

Kai-Yin Fok†, Chi-Tsun Cheng, and Chi K. Tse
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
Email: †zerofky@gmail.com

***Abstract*—A refinement process for nozzle path planning in 3D printing is presented in this paper. The nozzle path planning problem is formulated as an undirected rural postman problem (URPP). Based on the unique characteristics of URPP in 3D printing applications, a new refinement process is proposed to shorten the processing time of a conventional URPP solver. Performances of the proposed refinement process is evaluated using computer simulations. Simulation results show that when comparing with other URPP solvers, solvers with the proposed process have shorter processing time and can provide solutions with shorter transition length.**

***Index Terms*—Additive manufacturing, 3D printing, URPP, optimization.**

I. Introduction

In a typical fused deposition modeling (FDM) based 3D printing process, the computer-aided design (CAD) file of a 3D model is fed into a slicer software to break it down into multiple thin layers. Each layer of the model will then be decomposed into print segments and further be converted into control codes for the machining motions of the mechanical parts on a FDM machine. Among most off-the-shelf FDM machines, their printing nozzles move along the print bed while printing, and the extruder controls the flow rate of the filament. Molten plastic filaments, usually ABS or PLA, will be deposited via the printing nozzle onto the print bed to develop the print segments and construct the model layer by layer. To print a segment, the nozzle moves to the start point of the segment and traverses to its end. Meanwhile, the extruder injects filament toward the nozzle's reservoir and creates the required pressure. The molten filament is then pushed out of the nozzle and forms the print segment. Before the nozzle reaches the end of the segment, the extruder reduces the pressure gradually such that no extra filament will be deposited beyond the end of the current segment. The nozzle then moves to the start point of the next segment. The process repeats until all print segments on the current layer have been traversed. The print bed is then descended and the printing process of the next layer proceeds.

The total build time of a model is proportional to the time required for the printing nozzle to traverse all its segments including both print and non-print segments, which are also known as transitions. While the time spent on traversing print segments cannot be reduced, the total build time can be shortened by having shorter transitions. The routing of the printing nozzle can be formulated as an undirected rural postman problem (URPP), which was first introduced by Orloff in [1]. The objective is to find a minimum cost route, in terms of time or distance, which can traverse all the required edges $E_r \subseteq E$ at least once, where E is the set of all segments in a model. URPP is proven to be NP-hard if $E_r \neq E$ [2].

Frederickson's algorithm is a well-known approach for solving URPP and has an approximation factor of 1.5 [3]. It is similar to the approximation algorithm proposed by Christofides in [4] that aims to solve traveling salesman problem (TSP). In [5], Muyldermans *et al.* utilized k–opt algorithm in general routing problems (GRP), where k–opt is originally a refinement algorithm for TSP. It demonstrated significant improvement to solution quality in GRP. Groves and Vuuren in [6] proposed heuristic-based algorithms for RPP. Their simulation results show that their algorithms can provide decent solutions to RPP with relatively low computational complexities. In [7], Hertz *et al.* showed that improved RPP solutions can be obtained, with reasonable processing durations, by applying 2–opt algorithm to solutions generated by Frederickson's algorithm mentioned above. Recently, Fok *et al.* formulated the nozzle motion planning problem as TSP and proposed a relaxation scheme for shortening the processing time without causing significant impact to the model build time [8].

Besides nozzle motion planning, extensive researches have been conducted on optimizing other aspects of additive manufacturing processes. Cheng *et al.* in [9] studied how fabrication accuracy, build time, and cost of a FDM process be affected by the orientation of the printing model. They showed that by optimizing the orientation of a model, the amount of supporting materials needed can be greatly reduced and thus can shorten the build time.

In this work, by exploiting the unique characteristics of print segments, a refinement process is proposed to shorten the processing time of nozzle motion planning in FDM processes. In the later section, we can also observe that the total length of transitions traversed by the printing nozzle can be shortened with the help of the proposed refinement process. The rest of the paper is arranged as follow. Section II provides the problem formulations, including the motion model of the printing nozzle and the objective function used in the optimization process. Brief descriptions on Frederickson's algorithm and k–opt, the RPP solver and the corresponding enhancement

algorithm adopted in this work, are given in Section III. The unique characteristics in 3D Printing are elaborated in Section IV. The proposed refinement process is explained in Section V. Performances of the proposed refinement process were evaluated using computer simulations. Simulation settings and the corresponding results are presented in Section VI. Finally, concluding remarks are given in Section VII.

II. Problem Formulation

In this section, the formulation of the nozzle motion planning optimization problem is presented. In a FDM process, structures inside each layer of a 3D model are constructed by massive numbers of print segments. The printing nozzle must traverse all print segments and deposit the required amount of molten plastic filament onto them. Movements of the printing nozzle among disjointed print segments on the same layer are regarded as transitions. The speed of the printing nozzle and the filament flow rate can affect the thickness of a print segment. This makes the total time for the printing nozzle to traverse all print segments almost a constant. In contrast, the printing nozzle can shorten its traverse time on transitions by achieving its maximum speed. The objective of the optimization problem is therefore to find a fast path that can traverse all print segments in a model, which can also be viewed as minimizing the time for the printing nozzle to traverse transitions.

A. Undirected Rural Postman Problem

In this work, the optimization problem is formulated as an undirected rural postman problem (URPP), which is an alternated version of the Chinese postman problem (CPP). In CPP, given a graph $G = (V, E)$, a postman is required to find a shortest or fastest route, which can visit all the edges E on graph G at least once. The problem is very similar to the well-known TSP, except the objective is to visit all the edges E instead of the vertices V. In URPP, the requirement is relaxed as the postman is only required to visit some required edges $E_{\mathrm{r}} \subset E$ on graph G at least once. Under the URPP formulation, print segments are the required edges E_{r} while transitions are the remaining edges $E \setminus E_{\mathrm{r}}$. For a set of print segments E_{r}, let V_r denotes the set of vertices corresponding to the edges in E_{r}, a graph G can therefore be constructed by adding a set of extra edges E_{e} that connect every pair of vertices in V_{r}, such that $G = (V_{\mathrm{r}}, E = E_{\mathrm{r}} \cup E_{\mathrm{e}})$. The objective is therefore to find a fast tour that can traverse all E_{r} on G at least once.

B. Cost function

Since the total time needed by the printing nozzle to traverse all the print segments in the model is a constant, in the optimization process, only the cost associated with transitions will be considered. The cost function $c(v_i, v_j)$ represents the required time for a printing nozzle to traverse a transition (v_i, v_j) from vertex v_i to vertex v_j. The cost function consists of two components, i.e.,

$$c(v_i, v_j) = t_{\mathrm{d}}(v_i, v_j) + t_{\mathrm{r}}(v_i, v_j) \tag{1}$$

Here, $t_{\mathrm{d}}(v_i, v_j)$ represents the time required for the nozzle to traverse a transition (v_i, v_j), which can be obtained using the motion model of the nozzle. The second component $t_{\mathrm{r}}(v_i, v_j)$ denotes the time required by the extruder to perform retraction.

1) Motion model: In this work, we adopt the nozzle motion model in [8] for calculating $t_{\mathrm{d}}(v_i, v_j)$ in (1). Such model considers the acceleration and deceleration of the nozzle and thus allows its velocity to be varied while traversing an edge. For the print segments, during acceleration and deceleration of the nozzle, consistency on the thickness of the depositing filament is assumed to be controlled by adjusting its flow rate via the extruder. Assuming the printing nozzle can stop precisely at the coordinates as instructed, the cost (i.e., time) for the printing nozzle to traverse a transition can be obtained with the corresponding triangular and trapezoidal velocity profiles [10].

2) Retraction: The term *strings* refer to those extra filaments dripped out from the nozzle when it traverses across disjoint parts. Most FDM machines can alleviate such problem by performing *retraction* [11] before a traverse, which involves withdrawing solid filament in order to create a negative pressure at the reservoir inside the printing nozzle. The time spent on retraction is denoted as t_r, which is a function of the amount of filament being withdrawn and the corresponding withdraw rate.

III. Nozzle Path Planning Optimization

The proposed refinement process is a customized k–opt algorithm that operates on solutions obtained using Frederickson's algorithm. In this section, both Frederickson's and k–opt algorithms will be elaborated.

A. Frederickson's Algorithm

Frederickson's algorithm was first proposed in [3]. It is a widely-adopted construction algorithm for URPP which shows high similarities to the Christofides' algorithm in TSP [4]. Frederickson's algorithm begins with a given undirected and connected graph $G = (V, E)$ and a set of required edges E_{r}. A minimum spanning tree (MST) is then constructed to connect all the edges in E_{r}. Those new edges introduced in the MST construction process are forming a new set E_{mst}. A minimum perfect matching is then conducted on the sumset $E_{\mathrm{r}} + E_{\mathrm{mst}}$ to connect vertices with odd degrees. The extra edges added in the matching process are then forming another new set E_{matching}. An Eulerian tour can then be found in the sumset $E_{\mathrm{r}} + E_{\mathrm{mst}} + E_{\mathrm{matching}}$. Consecutive edges on the tour, which are both not in E_{r}, are replaced with shortcuts to further optimize the tour.

B. k–opt Algorithm

k–opt algorithm was first designed to further optimize sub-optimal TSP solutions [12]. It was modified by Hertz in [7] to enhance solutions of URPP. In k–opt algorithm, a given tour is broken down into k fragments and reconnected using different feasible combinations. The tour is updated whenever an improvement is found. The process repeats until

no further improvement can be achieved. There are two general implementations of k–opt algorithm. k–opt$_1$ updates the tour immediately whenever an improvement is found and starts a new iteration. k–opt$_2$ scans through all available combinations in the current iteration and updates the tour using a combination with the best improvement. In this work, both implementations of 2–opt were included in the simulations to study the performances of the proposed refinement process versus different configurations. Note that k–opt algorithms with higher orders were not considered due to their high computational complexities.

IV. Unique Characteristics in 3D Printing

In this section, some unique characteristics of URPP in 3D printing application are explained, which provide insights to the development of the proposed refinement process.

A. Zero-Cost Transitions

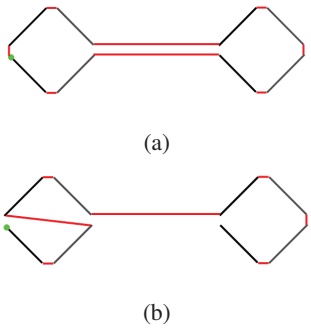
In the formulation, the cost of an edge is denoted as $c(v_i, v_j) \geq 0, \forall v_i, v_j \in V$ and $c(v_i, v_j) = c(v_j, v_i)$, which is a function of edge length and the velocity profile of the printing nozzle. Therefore, a transition joining vertices v_i and v_j with $c(v_i, v_j) = 0$ implies the vertices (and the corresponding print segments) are connected. While edges with zero cost rarely exist in ordinary URPP, it is quite common under the aforementioned formulation as curves in a model are usually constructed by multiple short print segments due to resolution limits. Computational complexity of the optimization process can be greatly reduced by giving them lower priorities in the calculation.

B. Starting and Ending Vertices

The goal of an ordinary URPP is to find a fast Euler tour that can traverse all the required edges E_r in G, and the path will return to the starting point. Such requirement on the starting and ending vertices is not necessary for 3D printing as the printing nozzle can start printing the next layer once all print segments in the current layer have been built. The illustrative example in Fig. 1 shows that there are rooms for further optimization by not returning the printing nozzle to its starting location.

V. Proposed Method

In this paper, a refinement process to the nozzle motion planning problem is developed by exploiting the unique characteristics mentioned in Section IV. The process comprises two major parts. The first part of the process introduces an extra rule to k–opt$_2$ when exploring the combinations for tour breaking. Before executing the k–opt$_2$ algorithm, a priority list of transitions on a given tour is created, where transitions with higher costs are given with higher priorities. During the searching operation of k–opt$_2$, transitions with higher priorities will be evaluated first. Such design is to address the fact that curvy lines in a model are assembled with large numbers of short print segments and zero-cost transitions. It is very unlikely for those zero-cost transitions to be replaced with better options in k–opt$_2$ algorithm. Therefore, they are given lower priorities.



(a)

(b)

Fig. 1: Illustrations showing (a) the optimal solution for a general URPP and (b) the optimal solution in 3D printing. Here, the green dot is indicating the starting point while the black lines and red lines are representing the print segments and transitions, respectively.

The second part of the proposed refinement process changes the order of the nozzle motion plan by omitting the requirement on returning to the starting vertex (i.e., starting vertex $\neq$ ending vertex). The process starts with a given visiting cycle of the print segments T that includes all edges in E_r, and the starting coordinate v_s of the current layer that is the current location of the printing nozzle. The cycle is then cut at a vertex on T which is closest to v_s and become a chain $C = \{(v_1, v_2), (v_3, v_4), \cdots, (v_i, v_{i+1}), \cdots, (v_{n-1}, v_n)\}$, where (v_i, v_{i+1}) are the corresponding vertices of the i^th print segment on the chain. It is worth to note that v_1 is the closest vertex on C to v_s. In the p^th iteration of the process, C is broken down into two subchains $C_1 = \{(v_1, v_2), ..., (v_{p-2}, v_{p-1})\}$ and $C_2 = \{(v_p, v_{p+1}), ..., (v_{n-1}, v_n)\}$. The order of the chain will be updated if the joint cost of C_1 and the flipped version of C_2 (i.e. $C_{2\text{ flipped}} = \{(v_n, v_{n-1}), ..., (v_{i+2}, v_{i+1})\}$) is lower than the original, such that $C \leftarrow \{C_1, C_{2\text{ flipped}}\}$. Otherwise C remains unchanged and the process proceed to its next iteration until all transitions have been evaluated.. This process reduces the associated cost of C by performing local searches iteratively.

VI. Simulations

A. Simulation Settings

Simulations were conducted to evaluate the performance of the proposed refinement process. Seven 3D models in [13] were selected as sample models in the simulations. The models were sliced using Cura [14] with its default settings. Retractions were enabled with 0.075 mm vertical hop and 4.5 mm retraction length. The filling density was set to be 10% of the total volume. The print plans were further optimized using the relaxation scheme proposed in [8]. Nevertheless, for comparison purposes, the path optimization module in each set of simulations is replaced with different combinations of Frederickson's algorithm, 2–*opt*, and the proposed refinement process.

TABLE I: Total transition length (TTL) and post-processing time (PPT) of print plans obtained using different path optimizers.

	Cura	Frederickson		Frederickson 2–opt$_1$		Frederickson 2–opt$_2$		Proposed	
Models	TTL(mm)	PPT(ms)	TTL(mm)	PPT(ms)	TTL(mm)	PPT(ms)	TTL(mm)	PPT(ms)	TTL(mm)
UltimakerRobot_support_2015	12834.33	4782.42	10620.31	5814.06	10435.17	5645.41	9969.54	5228.23	9543.88
TortureTestV2	91491.76	15720.59	63297.36	40109.00	62508.60	48097.44	62085.92	20749.65	61946.60
testModel	23210.27	606.19	15791.64	1172.35	14811.59	1270.36	14350.39	775.13	14293.65
dragon_65_tilted_large	32869.14	10116.87	22242.97	12233.80	21603.00	12273.11	21098.84	10803.08	20328.11
Debailey_x10	41818.06	5233.78	27064.91	7748.49	26216.87	7734.42	25647.93	5448.11	25222.57
ctrlV_3D_test	41268.51	27265.75	28140.21	109636.09	27376.74	125079.93	26898.81	44910.82	26274.58
3DHackerTest	69330.28	100156.28	40120.65	156250.15	38092.83	167719.19	37716.00	113032.40	37109.57

All resulting print plans were assessed using an open-source simulator GCodeAnalysor-1.0 [15]. The post-processing time required by the optimizers were the mean values obtained from 30 individual simulations executed on a computer with Window 8.1, Intel Core i7 processor, and 16 GB RAM. The simulation results are shown in Table I.

B. Results and Discussion

According to Table I, the proposed refinement process can always yield solutions with shorter total transition length (TTL) comparing to other methods under test. One possible reason is that the proposed process has considered the characteristics mentioned in IV-B in its design. By relaxing the restriction on starting and ending vertices in conventional URPP, the proposed process has more rooms for optimizing the solutions.

Regarding the post-processing time (PPT), it can be observed that the proposed process requires shorter time in optimizing the print plans of the selected models. The proposed refinement process can save an average of 31.47% and 34.50% in PPT when comparing with to Frederickson's algorithm followed by 2–opt$_1$ and 2–opt$_2$, respectively. Such improvement is mainly due to the imposition of the priority list as mentioned in IV-A.

It is noted that saving in PPT delivered by the proposed process for the model "UltimakerRobot_support_2015" is less significant than those in other models. This circumstance can be related to the special structure of the model, which consists mainly the shell and parallel filling lines. For more complicated models like "ctrlV_3D_test" that comprises more print segments and contains irregular polygons, the gap in PPT saving between the proposed process and Frederickson 2–opt$_2$ can be as large as 64.09%. This observation suggests that the proposed refinement process does scale well with the size of the optimization problem.

VII. Conclusion

In this paper, a refinement process for nozzle path planning in 3D printing is proposed. Its performances were evaluated using computer simulations. By incorporating the unique characteristics of the nozzle path planning problem in 3D printing, the proposed process can significantly reduce the post-processing time required for further optimizing print plans obtained from a path optimizer. Observable reductions are recorded when comparing the total transition length of its solutions with those generated by other algorithms under test. Simulation results also show that among understudying algorithms, the proposed refinement process demonstrates a relatively high scalability.

Acknowledgment

This work is supported by the Department of Electronic and Information Engineering, the Hong Kong Polytechnic University (Projects RU9D and G-YBKH).

References

[1] C. Orloff, "A fundamental problem in vehicle routing," *Networks*, vol. 4, no. 1, pp. 35–64, 1974.

[2] J. K. Lenstra and A. Kan, "On general routing problems," *Networks*, vol. 6, no. 3, pp. 273–280, 1976.

[3] G. N. Frederickson, "Approximation algorithms for some postman problems," *Journal of the ACM*, vol. 26, no. 3, pp. 538–554, Jul 1979.

[4] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," DTIC Document, Tech. Rep., Feb 1976.

[5] L. Muyldermans, P. Beullens, D. Cattrysse, and D. Van Oudheusden, "Exploring variants of 2-opt and 3-opt for the general routing problem," *Operations research*, vol. 53, no. 6, pp. 982–995, Dec 2005.

[6] G. Groves and J. Van Vuuren, "Efficient heuristics for the rural postman problem," *ORiON*, vol. 21, no. 1, pp. 33–51, Jan 2005.

[7] A. Hertz, G. Laporte, and P. N. Hugo, "Improvement procedures for the undirected rural postman problem," *INFORMS Journal on Computing*, vol. 11, no. 1, pp. 53–62, Feb 1999.

[8] K. Y. Fok, C. T. Cheng, C. K. Tse, and N. Ganganath, "A relaxation scheme for TSP-based 3D printing path optimizer," in *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Oct 2016, pp. 382–385.

[9] W. Cheng, J. Fuh, A. Nee, Y. Wong, H. Loh, and T. Miyazawa, "Multi-objective optimization of part-building orientation in stereolithography," *Rapid Prototyping Journal*, vol. 1, no. 4, pp. 12–23, 1995.

[10] N. Ganganath, C. T. Cheng, K. Y. Fok, and C. K. Tse, "Trajectory planning for 3D printing: A revisit to traveling salesman problem," in *2016 2nd International Conference on Control, Automation and Robotics (ICCAR)*, Apr 2016, pp. 287–290.

[11] Ultimaker, "How to fix stringing — Ultimaker: 3D printers," (Accessed: 2017-02-22). [Online]. Available: https://ultimaker.com/en/resources/19504-how-to-fix-stringing

[12] G. A. Croes, "A method for solving traveling-salesman problems," *Operations research*, vol. 6, no. 6, pp. 791–812, Dec 1958.

[13] Ultimaker, "GitHub - Ultimaker/CuraEngine," (Accessed: 2017-02-22). [Online]. Available: https://github.com/Ultimaker/CuraEngine

[14] Ultimaker, "Cura 3D printing slicing software," (Accessed: 2017-02-20). [Online]. Available: https://ultimaker.com/en/products/cura-software

[15] K. Y. Fok, "GCodeAnalysor-1.0 by kyfok - Thingiverse," (Accessed: 2016-11-9). [Online]. Available: http://www.thingiverse.com/thing:1870254