Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 53-2 (2020) 10037–10043

Additive Manufacturing Path Generation for Robot Manipulators Based on CAD Models ⋆

Ingrid Fjordheim Onstein * **Linn Danielsen Evjemo** ** **Jan Tommy Gravdahl** **

* *Department of Manufacturing and Civil Engineering, Norwegian University of Science and Technology, Gjøvik (e-mail: ingrid.f.onstein@ntnu.no)*

** *Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim (e-mail: (linn.d.evjemo, jan.tommy.gravdahl)@ntnu.no)*

Abstract: Traditional extrusion based additive manufacturing (AM) is realized using a 3 degrees of freedom (DOF), translation only, 3D printer. It then follows that the printer must be larger than the printed part. One way of enabling AM on a larger scale is to combine AM with robotics. By using a 6 DOF robot manipulator to extrude a fast-curing material, the workspace of the build would be greatly expanded. In addition, since the structures would no longer have to be built with the bottom-up or top-down approach which is necessary for most existing forms of AM, the flexibility of the building process would also increase. This could possibly reduce the need for support structures to the point of only relying of anchoring and stabilizing. In this paper, a method for generating a path for AM using robot manipulators that takes advantage of the robot's DOF is presented. The path is generated based on simple surfaces in CAD models. First, the surface is sampled and the samples are gathered in a point cloud. Then, a path is generated based on the point cloud. Three different approaches for generating a path are tested where the weighted greedy choice algorithm gave the most promising result. With this algorithm, printing along curved surfaces and in nonlinear paths are enabled.

Keywords: Additive manufacturing, path planning, robotic manipulators, robot programming, manufacturing systems, mechatronic systems.

1. INTRODUCTION

In the recent years, the advancements in additive manufacturing (AM) has increased rapidly and AM has become more available to the public. Traditionally, AM is realized with a layer-by-layer fabrication, where every layer is parallel to the $x-y$ plane of the build volume. It then follows that the 3D printer must be larger than the print. To enable large scale AM using traditional printers, the part must be printed in smaller sub-parts before being mounted together afterwards. Another way of enabling large scale AM is to combine AM with robotics (Evjemo et al., 2017). By using a 6 degrees of freedom (DOF) robot manipulator to extrude a fast-curing material, the workspace of the build could be greatly expanded. It would also be possible to increase the flexibility of the building process itself because the structure would no longer have to be built layer by layer using the top-down or bottom-up approach which is necessary for most existing forms of AM. This could reduce the need for support structures to the point of only relying on anchoring and stabilizing.

There has been a great deal of advancements within AM in the recent years. Livesu et al. (2017) present an overview of the AM processing pipeline from 3D model to 3D print, mostly concerning traditional AM. Evjemo et al. (2017) present an overview of state-of-the-art of large scale AM including a proof-of-concept experiment using a robot manipulator. MX3D (2019) and Stratasys (2019) are both commercial companies that present novel use of AM using robot manipulators, MX3D for large scale printing and Stratasys for removing the need of support structures. Within the subject of AM with non-planar contours, Zhang et al. (2016), Alsharhan et al. (2017) and Shembekar et al. (2019) show promising results. The resulting parts has improved material properties compared to planar contours. There has also been some recent development with using arc-welding techniques for AM, so-called WAAM. This is attracting interest from the manufacturing industry because of their potential to fabricate large metal components with low cost and short production lead time. A review of WAAM is presented in Pan et al. (2018). Evjemo et al. (2019) present experiments assessing the feasibility of large scale AM of metallic materials by arc welding.

Urhal et al. (2019) present a review of robot assisted AM, including a discussion on the difference in information flow compared to traditional AM. In Onstein (2017), the author

⋆ The work reported in this paper was based on activities within centre for research based innovation SFI Manufacturing in Norway, and is partially funded by the Research Council of Norway under contract number 237900.

10.1016/j.ifacol.2020.12.2724

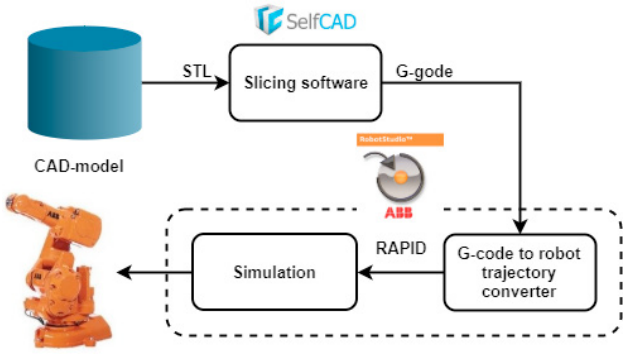


Fig. 1. Information flow of AM with 6 DOF robot using existing software

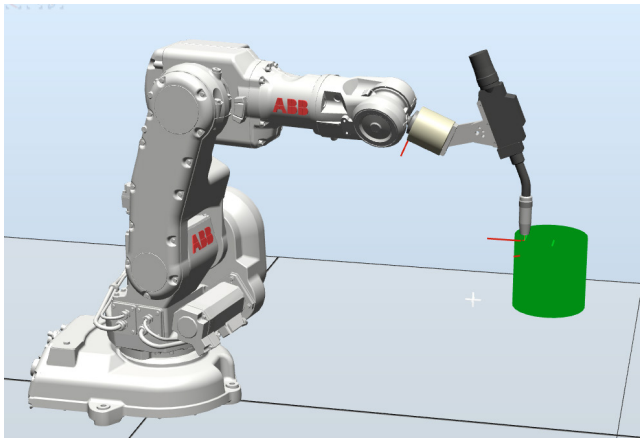


Fig. 2. Result of simulation of printing a cylinder in RobotStudio

investigated using existing software for AM with robot manipulators. The process pipeline, which is similar to the one presented in Urhal et al. (2019), is represented in figure 1. For slicing and generation of G-code, SelfCAD was used. To convert G-code into a robot trajectory, ABB RobotStudio and its Machining PowerPac was used. Machining PowerPac is an add-in to RobotStudio that includes a CAM converter (ABB, 2019). The CAM converter can convert G-code into RAPID which is the programming language used to program ABB's industrial robots. The method was verified using simulations and a proof-of-concept experiment on a physical robot. The simulations was realised using RobotStudio. After generating the trajectories using RobotStudio and the Machining PowerPac, the trajectories can be simulated within the same environment. An illustration of the simulation can be seen in figure 2. An experiment was also performed on an ABB IRB140 robot manipulator. No extrusion was tested in the experiment. Pollak et al. (2018) and Ribeiro et al. (2019) has also investigated using existing software for AM using robot manipulators.

There are both advantages and disadvantages with using existing solutions. The clear advantage is that the method use existing solutions for both traditional AM and robot control. It only takes a few minutes to generate a complete robot trajectory based on a CAD model. Furthermore, slicers and robot control environments both come with powerful visual tools. This makes it easy to modify parameters and simulate the process in very intuitive and interactive environments. The main disadvantage with the method is that the generated trajectory is intended for a system with 3 DOFs. The motivation for using a robot manipulator for AM is to take advantage of its 6 DOFs to enable paths that is not feasible with only 3 DOFs. When using existing solutions, AM is realised with a robot manipulator, but the CAD model might as well have been printed using a traditional 3D printer.

In this paper, a method for generating paths for AM based on STEP models that takes advantage of the robot manipulators 6 DOF is presented. The method enables printing along non-linear paths and in overhang, i.e. a part of a structure that stick out over the lower level of the build without direct vertical support. The method consists of generating a point cloud based on a STEP model and generating a path based on the generated point cloud. Three different approaches for generating the path based on the point cloud is evaluated. The work presented in this paper is part of a project that seeks to enable large scale AM using robot manipulators. The presented method is based on the assumption of depositing a fast curing material that enable mid-air printing. Since this is a novel method, only surfaces of CAD models are considered to simplify the task. Further work will include expanding the method into handling more complex 3D models.

This paper consists of 4 sections. After this introduction, section 2 present a method for generating paths for AM that seeks to exploit the 6 DOF a traditional robot manipulator holds. Section 3 presents the results and discuss the proposed method that is tested on two different CAD models. Finally, there is a closing section with some concluding remarks in section 4.

2. AM PATH GENERATION

Input to the path generation system is a CAD model in the STEP format. STEP is a data exchange file format defined in ISO (2002) that can represent a 3D object and related information. As opposed to the commonly used Stereolithography (STL) format in 3D printing which is an approximation of the surfaces of the model, STEP contains an accurate description of the surfaces. More specifically, it is a face on the CAD model that is used as input. A face is a surface that is bounded by a set of edges (Stroud, 2006). Since this is a novel approach, only the faces of the CAD model is considered. Further research will look into converting this method into considering 3D solids.

The system for generating a path for AM consists of two main parts; sampling the desired object and generating the path based on the samples. The system is implemented using Python and is run as a macro in FreeCAD. FreeCAD is an open-source CAD software that is highly customizable. Its functionality can be accessed and extended using Python (The FreeCAD Team, 2019). Macros are a convenient way to reproduce complex actions in FreeCAD. Actions recorded in a macro constitutes a list of Python function calls of the FreeCAD API. They can be further extended with custom Python logic to create complex scripts.

2.1 Generating point cloud

The surface $S \in \mathbb{R}^3$ is parametrized by a system on curvilinear coordinates given as

$$\mathbf{r}(u,v) = (x(u,v), y(u,v), z(u,v)) \tag{1}$$

with u and v being real variables. The normal to S is by definition a normal to a tangent plane, given by the cross product of the partial derivatives given as

$$\mathbf{n} = \frac{\partial r}{\partial u} \times \frac{\partial r}{\partial v} \tag{2}$$

The first step in the path generation system is to sample the face and generate a point cloud. Each sample is stored in a custom `Sample` object. This object contains reference to the FreeCAD sub object the sample belongs to, the curvilinear coordinates (u, v) and its corresponding Cartesian coordinates (x, y, z) as well as the normal vector $\mathbf{n}$. The sampling is realised by iterating through the curvilinear coordinates with a given step length in both u and v direction, namely α_u and α_v. All valid samples are stored and added to the point cloud. The sampling is given in algorithm 1.

Algorithm 1 Sampling sub object

Input: Desired distance between samples in each direction, d_u^* and d_v^*
Output: Point cloud

1: $\alpha_u = d_u^*$, $\alpha_v = d_v^*$
2: Set u and v max and min
$(u_{min}, u_{max}) = (\texttt{pRange(0)}, \texttt{pRange(1)})$
$(v_{min}, v_{max}) = (\texttt{pRange(2)}, \texttt{pRange(3)})$
3: Choose starting point
$(u_c, v_c) = (u_{min}, v_{min})$
4: **while** $u_c < u_{max}$ **do**
5: **while** $v_c < v_{max}$ **do**
6: Algorithm 2: Calibrate step length α_v given d_v^*
7: $v_c = v_c + \alpha_v$
8: $p_{c_{xyz}} = \texttt{valueAt}(u_c, v_c)$
9: **if** $p_{c_{xyz}}$ `isInside` face **then**
10: Add sample to point Cloud
11: **end if**
12: **end while**
13: $v_c = v_{min}$
14: Algorithm 2: Calibrate step length α_u given d_u^*
15: $u_c = u_c + \alpha_u$
16: $p_{c_{xyz}} = \texttt{valueAt}(u_c, v_c)$
17: **if** $p_{c_{xyz}}$ `isInside` face **then**
18: Add sample to point cloud
19: **end if**
20: **end while**

To sample the sub object, the user sets the desired distance (in millimetres) between each sample in both u and v direction, d_u^* and d_v^*. The actual distance between two samples can be calculated using the following equation

$$d_k = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{3}$$

where (x, y, z) is the Cartesian coordinate. The sub object is sampled with step length α_u and α_v given in the curvilinear coordinate system and the desired distance between samples d_u^* and d_v^* is given in the Cartesian coordinate system. It was therefore necessary to calculate the step lengths, α_u and α_v, that results in the desired distance d_u^* and d_v^*. The method used for calculating the step lengths, from here on referred to as calibration, is inspired by line search which is a basic iterative approach to find a local minimum of an objective function which is described in Nocedal and Wright (2006). The method for calibrating α_v is given in algorithm 2. Calibrating α_u is realised using the same algorithm by switching v and u in step 6. t is the tolerance given by the user to adjust the accuracy of the calibration. There are three FreeCAD commands used in the sampling algorithm; `pRange`, `valueAt` and `isInside`:

Algorithm 2 Calibrate step length α_v

Input: (u_c, v_c), d_v^*
Output: α_v

1: Set desired tolerance t
2: $\alpha_v = d_v^*$
3: Calibrated is false
4: **while** not calibrated **do**
5: $p_{1_{xyz}} = (x_1, y_1, z_1) = \texttt{valueAt}(u_c, v_c)$
6: $p_{2_{xyz}} = (x_2, y_2, z_2) = \texttt{valueAt}(u_c, v_c + \alpha_v)$
7: $d_k = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$
8: **if** $d_v^* > d_k + \frac{d_k}{t}$ OR $d_v^* < d_k + \frac{d_k}{t}$ **then**
9: $\alpha_v = \alpha_v \frac{d_v^*}{d_k}$
10: **else**
11: Calibrated is true
12: **end if**
13: **end while**

- `pRange` is the parameter range of the selected face containing the minimum and maximum value of u and v, and is given as an array with the four elements.
- `valueAt` takes the curvilinear coordinates (u, v) and returns the corresponding Cartesian coordinate (x, y, z).
- `isInside` checks if a given point lies on the sub object with a given tolerance. All samples that are found to lie on the face is added to the point cloud.

2.2 Path Algorithms

Given the point cloud, the next main step is to generate the path. Due to the fact that it is difficult to start and stop material extrusion, two requirements for the path are outlined. Firstly, the path should visit a point only once. Secondly, the path should not be self-intersecting. This problem shows resemblance to the Travelling Salesman Problem (TSP). TSP is described as a salesman wishing to make a tour visiting each city exactly once and finishing at the city he starts from with the total cost of the tour being minimal. It was therefore decided to test a TSP-solver for generating a path. Two more algorithms was tested, namely greedy choice and weighted greedy choice. The greedy algorithm was chosen because it is quick and easy to implement. The weighted greedy algorithm was implemented as an extension to the non-weighted greedy algorithm. The greedy algorithm makes the locally optimal choice in the hope that this choice will lead to a globally optimal solution. The weighted greedy algorithm differs in the way that a weight is added to the alternatives to influence the local optimal choice. Both algorithms

and TSP are described in Cormen (2009). TSP is NP-hard, meaning no polynomial-time algorithm has yet been discovered nor proven to exist to solve the problem. It is therefore expected that generating a path using TSP will require more time compared to the greedy choice algorithms.

Greedy choice
For the greedy algorithm, the locally optimal choice was set to be the nearest sample to the current one. The distance was calculated using equation 3. The algorithm is presented below.

Algorithm 3 Greedy choice algorithm

```
Choose starting sample
while there are unvisited samples do
    Calculate distance to the other samples
    Make greedy choice
    Update current sample
    Add sample to path
end while
```

Weighted greedy choice
As mentioned earlier, the weighted greedy choice is strongly related to the greedy choice. The difference is that a weight is added on samples satisfying a given weighting criteria. In this case, it was desired that the path followed a given direction to the end before changing direction. Direction here is referring to a direction in parameter coordinates. This was implemented by adding weights to v if u was the desired direction and the other way around. The algorithm is presented below.

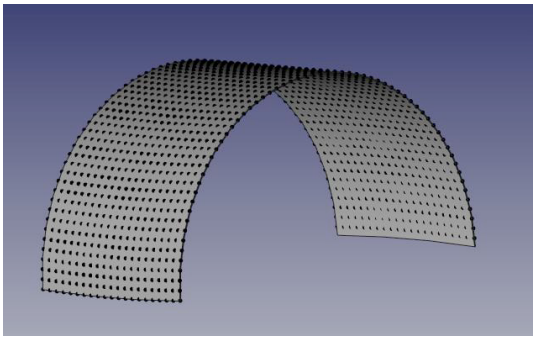
Algorithm 4 Weighted greedy choice algorithm

```
Choose starting sample
while there are unvisited samples do
    Calculate distance to the other samples
    if weighting criteria is satisfied then
        Add weight on distance
    end if
    Make greedy choice
    Update current sample
    Add sample to path
end while
```

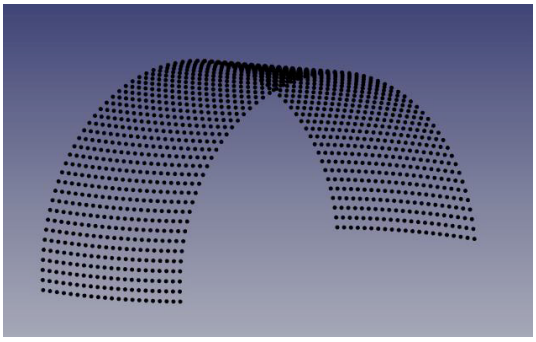
TSP
Like the two aforementioned algorithms, distance between the samples is used for picking the next sample for TSP as well. For generating the path based on TSP, a TSP solver was used. The solver used was inspired by Pedroso et al. (2020) and the source code is taken from Pedroso and Kubo (2018). The solver was modified to calculating distances in 3D instead of 2D using equation 3. The solver was tested and verified in Onstein (2018). The overall TSP-based algorithm using the TSP-solver is presented below.

Algorithm 5 TSP-based algorithm

```
Calculate distance matrix
Create greedy tour
Calculate length of tour
Local search starting from greedy tour
```



(a) Point cloud With CAD model.



(b) Point cloud without CAD model.

Fig. 3. Resulting point cloud from sampling a curved rectangular surface with $d_u^* = d_v^* = 10\ mm$.

3. RESULTS AND DISCUSSION

The path generation method was tested on two different CAD models; a curved rectangular surface and a propeller blade. The models were chosen to demonstrate paths for AM that would not be possible with only 3 DOF. The curved surface was chosen to demonstrate how a path for printing in overhang can be achieved and the propeller blade was chosen to demonstrate printing along non-linear paths.

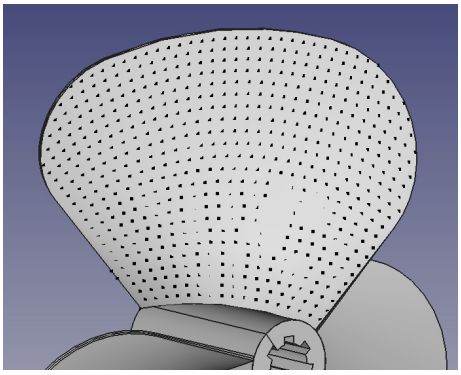
3.1 Sampling

Both models were sampled with a distance between the samples equal to $d_u^* = d_v^* = 10.0\ mm$ in both u and v direction. The results of sampling the curved surface can be seen in figure 3, and of the propeller blade in figure 4.

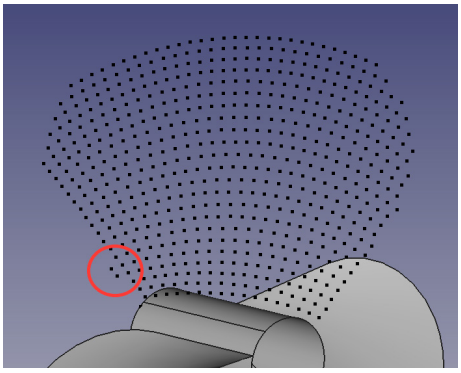
The sampling works well for both the curved surface and the propeller blade. The shape of the surfaces is well captured. In the point cloud of the propeller blade, there are two samples sticking out on the left side of figure 4b, highlighted with the red circle. These two points are a result of the chosen step length and will likely affect the path generation that is to come.

3.2 Path generation

The first algorithm that was tested was the greedy choice. The result from testing the algorithm on the curved surface can be seen in figure 5. In figure 5b it can be seen that the path is self-intersecting. To avoid a thick layer of material in the intersection, the material extrusion will have to be stopped and started again. This is a challenge with material extrusion and an intersecting path should

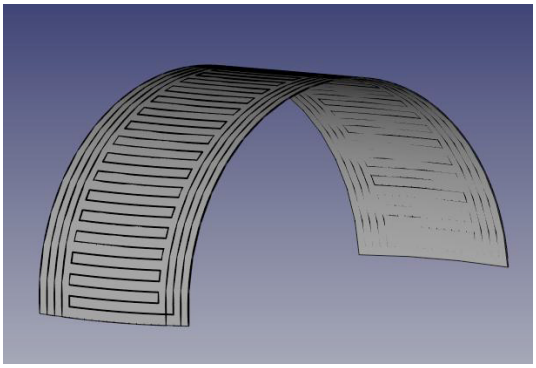


(a) Point cloud With CAD model.

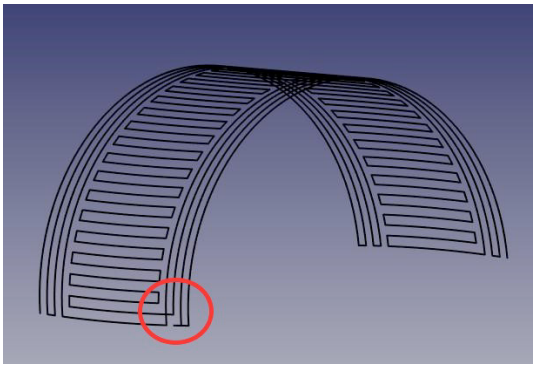


(b) Point cloud without CAD model.

Fig. 4. Resulting point cloud from sampling a propeller blade with $d_u^* = d_v^* = 10\ mm$.



(a) With CAD model.



(b) Without CAD model.

Fig. 5. Path generated using greedy choice algorithm on curved surface.

therefore be avoided. Results of running the greedy choice algorithm on the propeller blade can be seen in figure 5.9 in the previous work by the author (Onstein, 2018). Like the results of using the algorithm on the curved surface, the generated path is self-intersecting.

The next algorithm to be tested was the weighted greedy choice algorithm. The weights was added on the u-parameter to guide the path along the v-direction, resulting in movements from side to side. Results of the weighted greedy choice on the curves surface can be seen in figure 6. This path is moving from side to side and is without any self-intersections. The resulting path is also enabling printing in overhang. Figure 7 shows the weighted greedy choice on the propeller blade. In section 3.1, two samples sticking out on the left side was highlighted and it was pointed out that this was likely to influence the path generation. The result of those two samples can be seen in figure 7b. The path takes a longer turn on the left side before continuing on to the right. Despite this, the path for the propeller blade enable printing along non-linear curves. Both paths generated for the curved surface and for the propeller blades using the weighted greedy choice algorithm are paths that can be realised using a 6 DOF robot manipulator, but not with a traditional 3 DOF 3D printer.

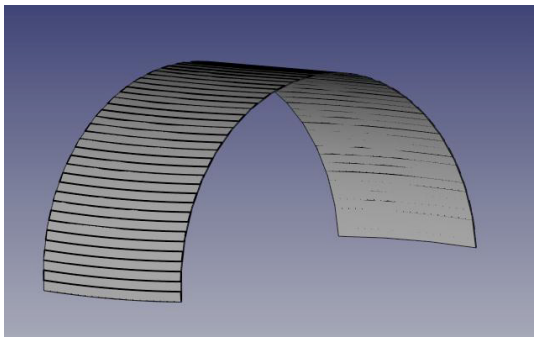
Figure 8 shows the result of the TSP-based algorithm on the propeller blade. This path is moving from side to side and up and down in a pattern that looks more like a maze than a path for AM. It is worth mentioning that the path is not self-intersecting. With an ideal, fast curing, material this path might have been plausible, but in all practical aspects it is not. Testing the TSP-based algorithm on the curved surface gave similar results as for the propeller blade. Another remark of using the TSP-based algorithm is that it took a long time to generate the path compared to the two greedy choice algorithms. This was expected as TSP is NP-hard.

In the authors previous work (Onstein, 2018), sampling and running the different algorithms was also tested on more than one surface at the time with a CAD model resembling a shoehorn. This revealed some issues regarding the sampling in the seam between surfaces that had implications on the path generation. More results and discussion around this can be found in the aforementioned work.

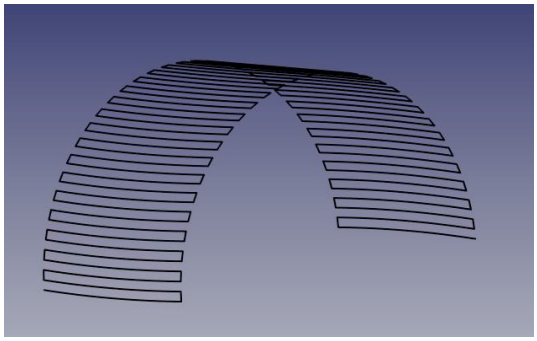
4. CONCLUSION

The advancements in AM has increased rapidly in recent years, especially using traditional 3D printers. There has been some research regarding AM using robot manipulators. Most of them use existing software for traditional 3 DOF printers and convert it into robot code. This paper present a method for generating paths for AM that takes advantage of the freedom in both position and orientation that a 6 DOF robot manipulator offers. Paths generated on a point cloud using a weighted greedy algorithm gave the most promising result. Through the presented method, both printing in overhang and along non-linear paths are realised.

In this paper, the paths have been generated, but not simulated or tested on a physical robot. Future work consists of verifying the generated paths through simulations

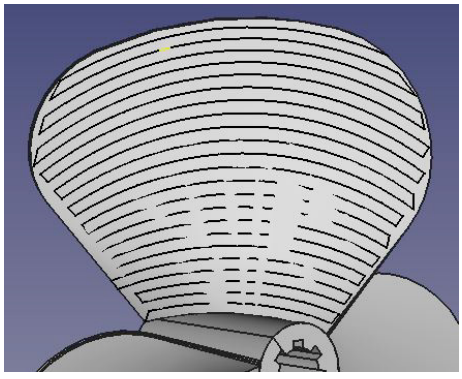


(a) With CAD model.

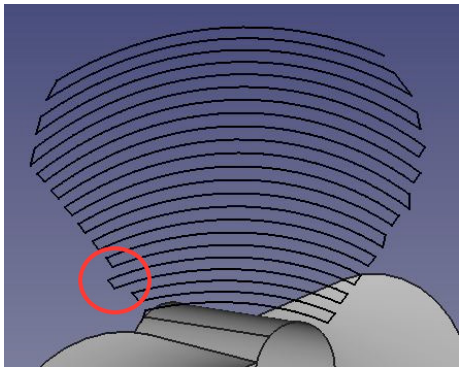


(b) Without CAD model.

Fig. 6. Path generated using weighted greedy choice algorithm with u-weighting on curved rectangular surface.

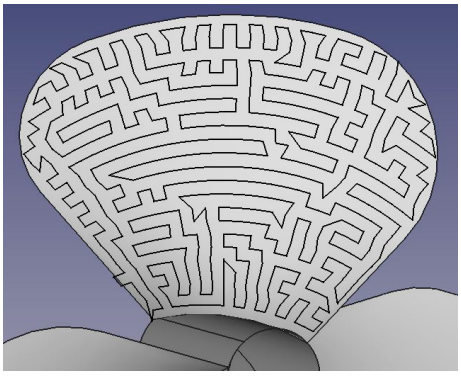


(a) With CAD model.

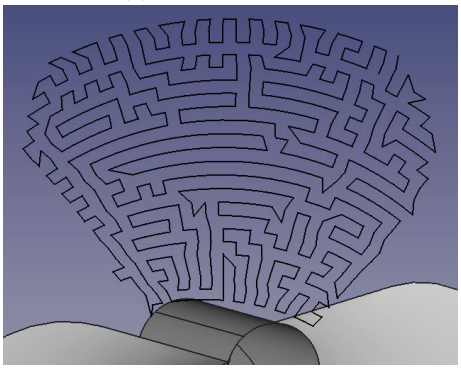


(b) Without CAD model.

Fig. 7. Path generated using weighted greedy choice algorithm with u-weighting on propeller blade.



(a) With CAD model.



(b) Without CAD model.

Fig. 8. Path generated using TSP on propeller blade.

and experiments. The presented method only considers the faces in the CAD model and not 3D solids. Further work will be to investigate how this method can be adapted to handle solids as well as faces. Furthermore, the problem of tool control and controlling material extrusion has not yet been addressed which is also a prospect of future work.

ACKNOWLEDGEMENTS

Associate professor Oleksandr Semeniuta at the Norwegian University of Science and Technology is gratefully acknowledged for valuable input.

REFERENCES

ABB (2019). Robotstudio machining powerpac. `http://new.abb.com/products/robotics/application-software/machining/robotstudio-machining-powerpac`. [Online; accessed 17-September-2019].

Alsharhan, A.T., Centea, T., and Gupta, S.K. (2017). Enhancing mechanical properties of thin-walled structures using non-planar extrusion based additive manufacturing. In *ASME 2017 12th International Manufacturing Science and Engineering Conference collocated with the JSME/ASME 2017 6th International Conference on Materials and Processing*, V002T01A016–V002T01A016. American Society of Mechanical Engineers.

Cormen, T.H. (2009). *Introduction to algorithms*. MIT press.

Evjemo, L., Moe, S., Gravdahl, J., Roulet-Dubonnet, O., Gellein, L., and Brøtan, V. (2017). Additive manufacturing by robot manipulator: An overview of the state-

of-the-art and proof-of-concept results. In *In proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Limassol, Cyprus*.

Evjemo, L.D., Langelandsvik, G., and Gravdahl, J.T. (2019). Wire arc additive manufacturing by robot manipulator: Towards creating complex geometries. In *5th IFAC International Conference on Intelligent Control and Automation Science (ICONS)*. Belfast, Nothern Ireland.

ISO (2002). Industrial automation systems and integration - product data representation and exchange - part 21: Implementation methods: Clear text encoding of the exchange structure. Standard, International Organization for Standardization, Geneva, CH.

Livesu, M., Ellero, S., Martínez, J., Lefebvre, S., and Attene, M. (2017). From 3d models to 3d prints: an overview of the processing pipeline. *Computer Graphics Forum*, 36(2), 537–564.

MX3D (2019). Mx3d. `http://mx3d.com/`. [Online; accessed 02-October-2019].

Nocedal, J. and Wright, S.J. (2006). *Numerical Optimization*. Springer.

Onstein, I.F. (2017). Additive manufacturing by robot manipulator: A method for generating robot trajectory based on CAD model using existing solutions. Technical report, Department of Engineering Cybernetics, Norwegian University of Science and Technology. Specialization project.

Onstein, I.F. (2018). *An Additive Manufacturing Path Generation Method Based on CAD Models for Robot Manipulators*. Master's thesis, Department of Engineering Cybernetics, Norwegian University of Science and Technology.

Pan, Z., Ding, D., Wu, B., Cuiuri, D., Li, H., and Norrish, J. (2018). Arc welding processes for additive manufacturing: A review. In *Transactions on Intelligent Welding Manufacturing*, 3–24. Springer.

Pedroso, J.P. and Kubo, M. (2018). Tsp solver. `http://www.dcc.fc.up.pt/~jpp/code/py_metaheur/tsp.py`. [Online; accessed 20-May-2018].

Pedroso, J.P., Rais, A., Kubo, M., and Muramatsu, M. (2020). Tsp solver. `https://scipbook.readthedocs.io/en/latest/index.html`. [Online; accessed 05-May-2020].

Pollak, M., Torok, J., Zajac, J., Kocisko, M., and Teliskova, M. (2018). The structural design of 3D print head and execution of printing via the robotic arm ABB IRB 140. In *2018 5th International Conference on Industrial Engineering and Applications, ICIEA 2018*, 194–198. Institute of Electrical and Electronics Engineers Inc. doi:10.1109/IEA.2018.8387095.

Ribeiro, F.M., Pires, J.N., and Azar, A.S. (2019). Implementation of a robot control architecture for additive manufacturing applications. *Industrial Robot*, 46(1), 73–82. doi:10.1108/IR-11-2018-0226.

Shembekar, A.V., Yoon, Y.J., Kanyuck, A., and Gupta, S.K. (2019). Generating robot trajectories for conformal three-dimensional printing using nonplanar layers. *Journal of Computing and Information Science in Engineering*, 19(3). doi:10.1115/1.4043013.

Stratasys (2019). Stratasys demonstrators. `https://www.stratasys.com/demonstrators`. [Online; accessed 01-October-2019].

Stroud, I. (2006). *Boundary representation modelling techniques*. Springer Science & Business Media.

The FreeCAD Team (2019). Freecad. `https://www.freecadweb.org/`. [Online; accessed 08-October-2019].

Urhal, P., Weightman, A., Diver, C., and Bartolo, P. (2019). Robot assisted additive manufacturing: A review. *Robotics and Computer-Integrated Manufacturing*, 59, 335–345. doi:10.1016/j.rcim.2019.05.005.

Zhang, G.Q., Spaak, A., Martinez, C., Lasko, D.T., Zhang, B., and Fuhlbrigge, T.A. (2016). Robotic additive manufacturing process simulation-towards design and analysis with building parameter in consideration. In *Automation Science and Engineering (CASE), 2016 IEEE International Conference on*, 609–613. IEEE.