

Title:

**An Output-sensitive Algorithm for Generating Island Scanning Paths with Applications in SLM Additive Manufacturing**

Authors:

Ruiliang Feng, fengruiliang@sjtu.edu.cn, Shanghai Jiao Tong University

Lin Zhu, zhulin0728@sjtu.edu.cn, Shanghai Jiao Tong University

Peng Li, lipeng314159@126.com, Shanghai Aerospace Equipments Manufacturer Co., Ltd

Xiangzhi Wei, antonwei@sjtu.edu.cn, Shanghai Jiao Tong University

Keywords:

Additive Manufacturing, SLM, Path Planning, Scanning Strategy, Island Scanning

DOI: 10.14733/cadconfP.2019.227-231

Introduction:

Selective Laser Melting (SLM), a type of technology of Additive Manufacturing, is mainly used in manufacturing mechanical products with applications in the aeronautic, medical and automotive industry. Melted by the high power of the laser, a spot of melt powder will be transformed into a molten pool, which will be cooled into solid as the laser moves away. By repeating this process spot by spot, the SLM machine transforms one layer of powder into solid and repeat this process in a layer-by-layer manner.

For the path scanning strategies, there are mainly three types of patterns: unidirectional fill, bi-directional fill and island scanning. The path scanning strategies significantly influence the mechanic property of SLM part [3],[6],[8] Unsuitable scanning strategies can lead to defects on the SLM parts. Island scanning strategy is more generally used to avoid those defects [1],[4],[7],[9]. Lu et al. [5] investigated the influence of different island sizes for the fabrication of the Inconel-718 parts in SLM, Kruth et al. [4] found that shorter scanning vector could reduce the residual stress. Carter et al. [1] used island scanning to reduce the residual stress in SLM. Carter et al. [2] found microstructure of the grain structure of the SLM part fabricated with island scanning strategy was bimodal; Further, their MicroCT data suggests that cracks occurred in the fine-grained boundary regions.

However, the articles in literature that focuses on how to generate island scanning pattern efficiently are very sparse. In this paper, we develop an efficient and robust algorithm to generate the island scanning paths for SLM part. Our major contribution is an output-sensitive algorithm for generating the island scanning paths.

Main Idea:

Island scanning comply with the idea that divides the layer into many small isolated squares. Refer to Fig. 1(a), given two strips  $S_X^I$  and  $S_Y^{II}$ , where  $S_X^I$  is  $X^{\text{th}}$  strip directed along  $V$  and  $S_Y^{II}$  is  $Y^{\text{th}}$  strip directed along  $U$ , an island can be generated by the following manner: If  $X+Y$  is an odd number, then the interaction of  $S_X^I$  and  $S_Y^{II}$  is *Type I* island, otherwise it is of *Type II* island. In this paper, an island is denoted by  $(X, Y)$ .

Considering the above property, to generate island scanning paths, first, we obtain all the segments of intersection of contour  $C$  with lines directed along  $U$  and  $V$ . And then split each segment into candidate paths with the strips it crosses. However, not all these candidate paths are available.

For example, in Fig. 1(c), the candidate paths obtained from segment  $p_3p_4$  locates in  $(X+2, Y+1)$ ,  $(X+2, Y+2)$  and  $(X+2, Y+3)$  respectively, however only paths locate in  $(X+2, Y+1)$  and  $(X+2, Y+3)$  are filled with paths that directed along  $V$ , hence, path in  $(X+2, Y+2)$  is unavailable.

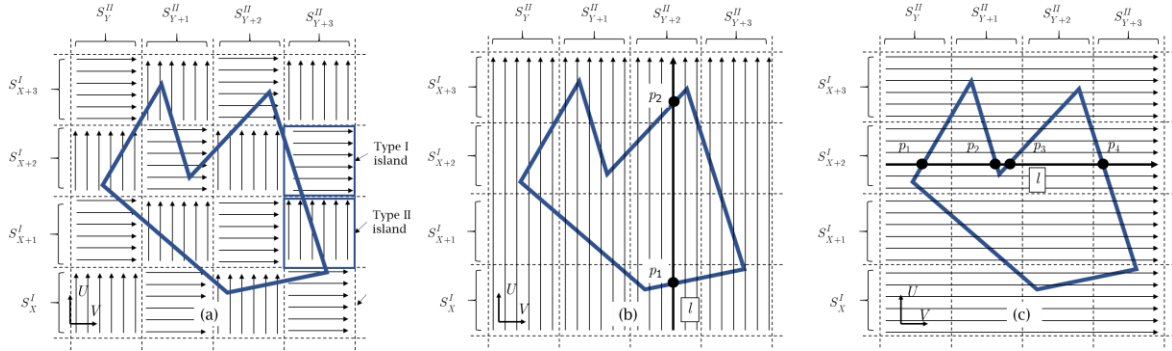


Fig. 1: Illustration of the generation of Island Scanning Strategy: (a) The target pattern. (b) The intersection of the contour with parallel lines directed along  $U$ , (c) The intersection of the contour with parallel lines directed along  $V$ .

#### The Details of the Algorithm

Given the contour of each layer, denoted by  $C$ , the full complete algorithm for generating the island paths for the interior of  $C$  is given below.

---

##### Algorithm Generate\_Island\_Paths ( $C$ )

---

**Input:** the set of boundary segments,  $\partial C$ .

**Output:** A compact and implicit table,  $Q$ , storing paths for islands.

```

1   $L := \emptyset$ ,  $Q := \emptyset$ ;
2  for each  $c \in \partial C$  do
3    for each line  $l$  crossed by  $c$  do
4      if  $l \notin L$  then
5         $L := L \cup \{l\}$ ;
6         $p := c \cap l$ ;
7         $P(l) := P(l) \cup \{p\}$ ; //  $P(l)$  is the set of intersection points of  $l$  with  $C$ 
8  for each  $l \in L$  do
9    if  $l$  directed along  $V$  then
10      $T := \text{Generate\_Type\_I\_Paths}(l)$ ;
11   else
12      $T := \text{Generate\_Type\_II\_Paths}(l)$ ;
13    $Q := Q \cup T$ 
14 Return  $Q$ ;
```

---

---

**Algorithm Generate\_Type\_I\_Paths ( $l$ )**


---

**Input:**  $l$ **Output:** a set,  $T$ , storing of Type I island paths.

```

1  $S := \emptyset$ ,  $i := 0$ ,  $X :=$  the index of the strip that  $l$  belongs to;
2  $P := \text{SORT}(P(l))$ ; // sort points along  $l$ 
3 while  $i < |P|$  do
4    $S := S \cup \{P_{i+1}\}$ ;
5    $i := i + 2$ ;
6 for each  $s \in S$ 
7   for each  $S_Y^H$  crossed by  $s$ 
8      $e := S_Y^H \cap s$ ;
9     if  $(X+Y)$  is an odd number then
10       $T(X, Y) = T(X, Y) \cup \{e\}$ ;
11 return  $T$ ;
```

---

Note that the Algorithm **Generate\_Type\_II\_Paths** ( $l$ ) is similar to Algorithm **Generate\_Type\_I\_Paths** ( $l$ ), the pseudo code of it is omitted for brevity.

### Complexity Analysis

Algorithm **Generate\_Island\_Paths**( $C$ ) has a total time complexity of  $O(K \log K + |C|)$ , where  $K$  is the number of laser paths in the interior of contour  $C$ , and  $|C|$  is the number of the boundary edges of  $C$ . The details of analysis are given below.

Line 1 takes  $O(1)$  time.

In Lines 2-7, the operation of computing the intersection of contour  $C$  with lines directed along  $U$  and  $V$ , which takes  $O(m + |C|)$  time, where  $m$  is the number of intersection points.  $m$  is smaller than twice of the number of island paths in the interior of contour  $C$ , hence it takes  $O(K + |C|)$  times,

Algorithm **Generate\_Type\_I\_Paths** ( $l$ ) takes  $O(k \log k)$  time, where  $k$  is the number of island paths on  $l$ . Due to  $|P(l)|$  cannot be larger than two times of  $k$ , the sorting of  $P(l)$  (line 2) takes  $O(k \log k)$  times; Lines 3-5 takes  $O(|P(l)|)$  time, i.e.  $O(k)$  time; lines 6-10 takes  $O(k)$  time. Algorithm **Generate\_Type\_II\_Paths** ( $l$ ) has a same time complexity with Algorithm **Generate\_Type\_I\_Paths** ( $l$ ). Therefore, Lines 8-13 in Algorithm **Generate\_Island\_Paths** ( $C$ ) totally takes  $O(K \log K)$  times.

In conclusion, the algorithm takes  $O(K \log K + |C|)$  times, i.e., the running time is proportional to the number of the output segments, which means that the algorithm is output-sensitive.

### Spinning Up the Island Scanning Pattern

Note that **Algorithm Generate\_Island\_Paths** ( $C$ ) only works for regular background patterns with vertical and horizontal lines, in order to compute the paths of contour  $C$  with a rotated island pattern (Fig. 2(a).) we need to adapt it as follows: first of all, we rotate the background pattern by an angle of  $\alpha$  in clockwise order (Fig. 2(b).); and then we rotate the contour  $C$  by an angle of  $\alpha$  in the clockwise direction (Fig. 2(c).), and then we use **Algorithm Generate\_Island\_Paths** ( $C$ ) to generate islands enclosed in the inter of  $C$ ; finally, we rotate  $C$  and the background pattern together by an angle of  $\alpha$  in the counterclockwise direction.

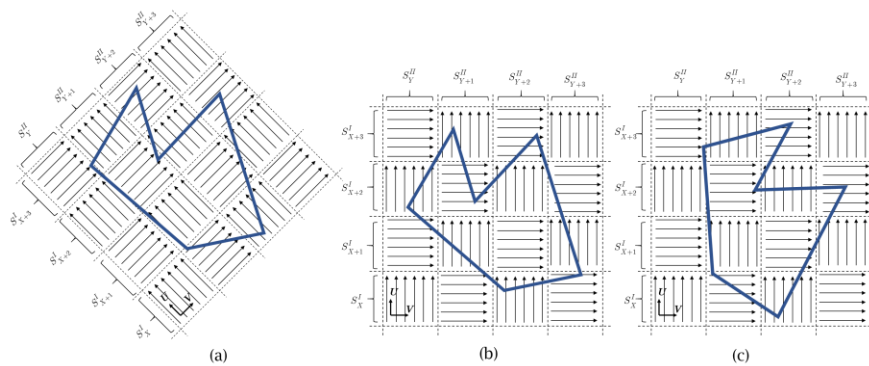


Fig. 2: Illustration of rotating island scanning paths by an angle of  $\alpha$ : (a) The target state, (b) The background pattern is rotated by an angle of  $\alpha$  in the clockwise direction, (c) The contour  $C$  is rotated by an angle of  $\alpha$  in the clockwise direction.

### Examples:

We have implemented this algorithm in C++ with Qt library and OpenGL API. We name our software as “Meshprint”. Refer to Fig.3., three models are simulated in “Meshprint”, the images in each figure contains the effects of loading models, generating support structures, and island scanning path planning.

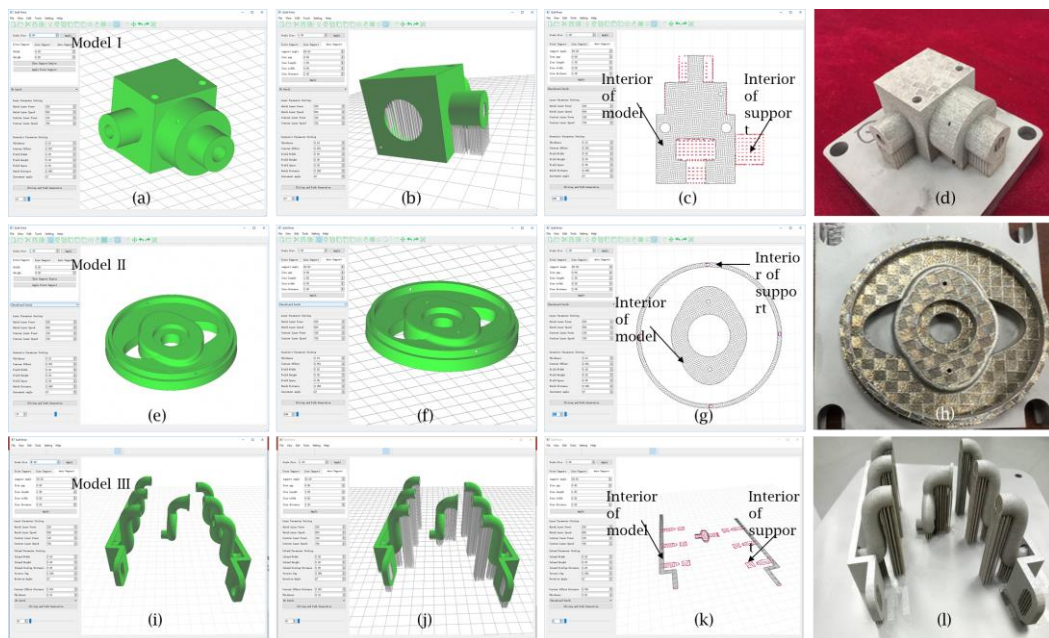


Fig. 3: The simulation of three models (Model I, II, III) using our self-developed “Meshprint” software: (a, e, i) The input models, (b, f, j) The supported models, (c, g, k) The inspections of paths of one layer, (d, h, l) The SLM 3D printed parts.

On a computer with Intel® Core™ i7-4770K @ 3.50GHZ and 8GB RAM, we compared the running time of “Meshprint” with the island scanning function of the path planning module of Materialise. Materialise is a company that specializes in developing software for Additive Manufacturing. It can be seen from Table 1 that “Meshprint” consumes less time, saving about 70%-80%.

<i>Test model</i>	<i>Dimension (mm×mm×mm)</i>	<i>Number of triangles</i>	<i>Number of layers</i>	<i>Consuming time of Magics</i>	<i>Consuming time of Meshprint</i>	<i>Save time</i>
Model I	60×71.5×44	32444	1467	24.00	6.5	72%
Model II	93×89×13	22384	400	9.0	1.3	85%
Model III	135×192.7×71.5	45176	2383	28.0	7.3	73%

Tab. 1: The comparison between the island path canning modules of Materialise and our self-developed “Meshprint” software.

### Conclusion:

Island scanning strategy has been shown to be an effective means in reducing the residual stress to product SLM part with less distortion and cracking. However, there is few pieces of literature and handouts of the commercial software to compute the island scanning path effectively for a large number of layers of a 3D printed model. In this paper, we propose an output-sensitive algorithm for generating the island scanning paths, and integration it into our software.

### References:

- [1] Carter, L. N.; Attallah, M. M.; Reed, R. C.: Laser powder bed fabrication of nickel-base superalloys: influence of parameters; characterisation, quantification and mitigation of cracking, Superalloys, 2012, 577-586. <https://doi.org/10.1002/9781118516430.ch64>
- [2] Carter, L. N.; Martin, C.; Withers, P. J.; Attallah, M. M.: The influence of the laser scan strategy on grain structure and cracking behaviour in SLM powder-bed fabricated nickel superalloy, Journal of Alloys and Compounds, 615, 2014, 338-347. <https://doi.org/10.1016/j.jallcom.2014.06.172>
- [3] DebRoy, T.; Wei, H. L.; Zuback, J. S.; Mukherjee, T.; Elmer, J. W.; Milewski, J. O.; Zhang, W.: Additive manufacturing of metallic components–process, structure and properties, Progress in Materials Science, 92, 2018, 112-224. <https://doi.org/10.1016/j.pmatsci.2017.10.001>
- [4] Kruth, J. P.; Deckers, J.; Yasa, E.; Wauthlé, R.: Assessing and comparing influencing factors of residual stresses in selective laser melting using a novel analysis method, Proceedings of the institution of mechanical engineers, Part B: Journal of Engineering Manufacture, 226(6), 2012, 980-991. <https://doi.org/10.1177/2F0954405412437085>
- [5] Lu, Y.; Wu, S.; Gan, Y.; Huang, T.; Yang, C.; Junjie, L.; Lin, J.: Study on the microstructure, mechanical property and residual stress of SLM Inconel-718 alloy manufactured by differing island scanning strategy, Optics & Laser Technology, 75, 2015, 197-206. <https://doi.org/10.1016/j.optlastec.2015.07.009>
- [6] Sames, W. J.; List, F. A.; Pannala, S.; Dehoff, R. R.; Babu, S. S.: The metallurgy and processing science of metal additive manufacturing, International Materials Reviews, 61(5), 2016, 315-360. <https://doi.org/10.1080/09506608.2015.1116649>
- [7] Shiomi, M.; Osakada, K.; Nakamura, K.; Yamashita, T.; Abe, F.: Residual stress within metallic model made by selective laser melting process, CIRP Annals-Manufacturing Technology, 53(1), 2004, 195-198. [https://doi.org/10.1016/S0007-8506\(07\)60677-5](https://doi.org/10.1016/S0007-8506(07)60677-5)
- [8] Tabernero, I.; Lamikiz, A.; Martínez, S.; Ukar, E.; Figueras, J.: Evaluation of the mechanical properties of Inconel 718 components built by laser cladding, International Journal of Machine Tools and Manufacture, 51(6), 2011, 465-470. <https://doi.org/10.1016/j.ijmachtools.2011.02.003>
- [9] Zaeh, M. F.; Branner, G.: Investigations on residual stresses and deformations in selective laser melting, Production Engineering, 4(1), 2010, 35-45. <https://doi.org/10.1007/s11740-009-0192-y>