第 58 卷 第 21 期/2021 年 11 月/激光与光电子学进展 研究论文

激光与光电子学进展

# 基于层间信息继承的金属增材制造扫描线填充算法

李慧贤[1]，吴陈浩[1]，马良[2*]

[1]西北工业大学计算机学院，陕西 西安 710072；

[2]西北工业大学材料学院，陕西 西安 710072

**摘要** 金属增材制造具有快速、无模具、自由成形复杂结构的特点，已经成功应用于航空、航天、模具、医疗等领域。随着制造零件的复杂程度和体积的不断增加，三维模型的数据量增大，数据处理所需要的时间大幅增加，尤其是路径规划所需的时间陡增，这已经成为制约该技术应用的主要瓶颈，亟需解决。为了减少路径规划所需的时间，基于增材制造连续两层之间轮廓相似这一基本事实，将三维模型切片得到的二维轮廓按特征进行分组，提出层间信息继承算法，充分利用上一层计算的填充路径信息，快速计算出当前层的填充路径。该算法无需计算每条扫描线与当前层众多轮廓环的交点，极大减小了路径填充的计算量，加快了填充速度。实验结果表明，该算法整体的计算效率明显高于传统路径填充算法，尤其对于等截面或截面连续变化的模型，该算法的加速效果尤为突出。

**关键词** 激光光学；增材制造；路径填充；层间信息继承；扫描线填充算法

**中图分类号** TG156 **文献标志码** A **doi**：10.3788/LOP202158.2114006

# Scan Line Filling Algorithm for Metal Additive Manufacturing Based on Information Inheritance between Layers

Li Huixian[1], Wu ChenHao[1], Ma Liang[2*]

[1]*School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi* 710072, *China*;

[2]*School of Materials, Northwestern Polytechnical University, Xi'an, Shaanxi* 710072, *China*

**Abstract** Metal additive manufacturing has been successfully applied in aviation, aerospace, mold, medical and other fields due to its characteristics such as rapid, mold-free, and free-form complex structure. As the complexity and volume of manufactured parts continue to increase, the amount of data in 3D models is increasing, and the time required for data processing has increased significantly, especially the steep increase in path planning, which has become the main bottleneck restricting the application of this technology. This problem needs to be solved urgently. In order to reduce the time required for path planning, based on the basic fact that the contours between two consecutive layers of additive manufacturing are similar, here the two-dimensional contours obtained by the slices of the three-dimensional model are divided into groups according to characteristics, and an innovative inter-layer information inheritance algorithm is proposed. Using the calculated filling path information of the previous layer, the filling path of the current layer can be quickly calculated. This algorithm does not need to calculate the intersections of each scan line and many contour loops of the current layer, which greatly reduces the calculation amount of path

**收稿日期**：2021-02-22；**修回日期**：2021-03-01；**录用日期**：2021-03-12

**基金项目**：国家重点研发计划(2018YFB1105303)、陕西省重点研发计划(2019ZDLGY11-01-01)

**通信作者**：*maliang@nwpu.edu.cn

filling and speeds up the filling. The experimental results have proved that the overall calculation efficiency of the proposed algorithm is significantly higher than that of the traditional path filling algorithm, especially for the models with a constant cross-section or continuous cross-section, and the acceleration effect of the proposed algorithm is more excellent.

**Key words** laser optics; additive manufacturing; path filling; information inheritance between layers; scan line filling algorithm

**OCIS codes** 140.1135; 350.3390; 200.3050

## 1 引 言

增材制造(Additive Manufacturing, AM)技术(也称为3D打印技术)是起源于20世纪80年代的一种逐层堆积的打印制造方法,与传统制造业相比,具有设计更加自由、制造更加简单的优势。2013年,在美国麦肯锡咨询公司发布的报告中,将增材制造技术列入决定未来经济的十二大颠覆技术之一[1]。该技术是集现代CAD/CAM(Computer Aided Design/Computer Aided Manufacturing)技术、激光技术、计算机数控技术、精密伺服驱动技术以及新材料等技术为一体的先进制造技术[2]。增材制造将零件模型进行分层切片,将材料按照切片信息的轮廓及打印路径进行逐层打印,直至打印出完整零件[3]。该技术已经在航空、航天、模具、医疗等领域取得了成功应用。

随着增材制造技术在各个行业的成功应用,增材制造模型的种类、复杂程度、尺寸和精度不断提高,路径填充算法所需处理的数据量越来越大。1 GB以上的立体光刻(Stereo Lithography, STL)三维模型越来越多,数据处理往往需要数个小时,严重影响工业生产。在金属增材制造模型数据处理的各个步骤中,最耗时的是路径填充过程。目前,大多数数据处理软件都是采用传统的路径填充算法,逐层地对轮廓进行填充,这个过程包含了大量的重复计算,加速路径生成已经成为增材制造行业的迫切需求。

增材制造的典型数据处理流程是利用STL三维模型先进行分层切片,再路径填充,最后后置处理,其中路径填充是增材制造数据处理过程的关键步骤[4-6]。该过程需要对分层切片得到的众多层的轮廓依次进行填充计算,不同的增材制造工艺需要不同的填充算法,扫描线填充[7]和轮廓偏置填充[8]是最典型的填充算法,除此之外,还有分形扫描方法[9]、分区域填充方法[10]等。甘泉[11]提出一种通用的扫描线多边形填充算法,采用了坐标变换、浮点数舍入策略等方法,可以有效地解决任意间距、任意倾角的扫描线对多边形的填充问题。Aiyiti等[12]提出动态调整平行扫描线间距的方法,通过分析层面轮廓线的几何特性,动态分配平行扫描线的间距,减少了扫描线间距相同带来的误差。黄雪梅等[13]提出了一种不必判别每条扫描线所在区域的分区扫描矢量生成算法,提高了扫描矢量生成算法的效率,同时优化了不同区域之间的路径,减少了空行程和加工耗时。Jin等[14]提出了将轮廓线偏置填充与分层变向往返填充方式相结合的填充路径。马良等[15]提出了基于信息继承思想的可连轮廓组快速提取算法,利用STL模型切片后相邻两层间的信息继承特性,提高了切片效率。

通过以上研究可以看出,国内外的研究人员对增材制造打印路径填充的研究较为广泛,对内部填充路径的规划研究较多,但是对轮廓路径填充过程的加速研究较少。尽管现有算法在一定程度上均能提高增材制造打印的制造效率和成型质量,但是它们都未充分考虑加速轮廓路径填充的问题。本文采用层间继承算法,对轮廓的填充过程进行加速,对于等截面或截面连续变化的模型,取得了较好的加速效果。

## 2 基于信息继承的路径填充算法

扫描线填充算法是用一组扫描线填满切片轮廓内部的实体区域,其实质就是计算一组扫描线与轮廓环的交点。该算法在增材制造路径规划中得到广泛使用,同时也是数控加工路径规划的主要算法。本文提出了基于层间信息继承的扫描线填充算法,该算法的主要思路是:假定每一层都使用扫描线填充算法进行填充,将模型分层后的层片按几何特征分为若干组,不同组的轮廓是不相似或不相等的,但每组内部任意两层轮廓都是相似或相等的。每组起始层片使用原扫描线填充算法进行填充,以供该组后续层片使用。利用同一组内任意两层层片轮廓相似的特性,使用前一层已生成的路径来生成后一层

的路径，不需要计算出每条扫描线与轮廓的交点，将交点连接成路径，减少了计算量，加快了算法的运行速度，有效地提高了路径填充的效率。

### 2.1　构建数据结构

针对基于信息继承的算法需求，本文提出了一种改进的数据结构模型。数据结构如图 1 所示，其中 Layer 表示层，LayerPart 表示层部件。

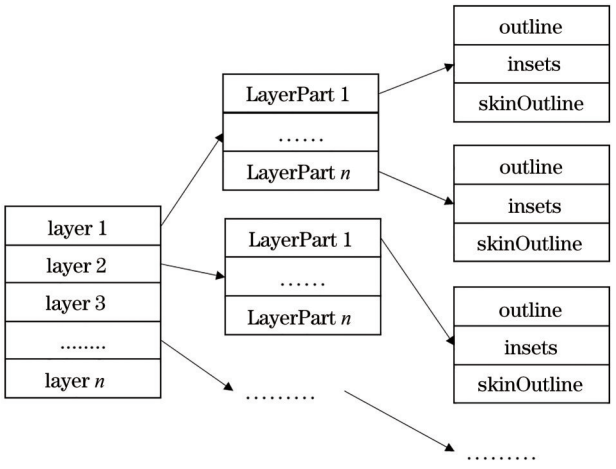


图 1　算法的数据结构

Fig. 1　Data structure of proposed algorithm

定义层、层部件、可连轮廓组三个结构体，使其分别用于存储层片、层部件和填充路径数据。

层结构体的定义如图 2 所示，其中 sliceZ 用存储该层的切片高度，printZ 用于存储该层的打印高度，parts 数组用于存储该层所有的层部件，result 用于存储该层的填充路径。

```
class SliceLayer
{
    int sliceZ;
    int printZ;
    vector<LayerPart>parts;
    Polygons result;
}
```

图 2　层的定义

Fig. 2　Definition of Layer

层部件结构体的定义如图 3 所示，其中 outline 用于存储该部件轮廓组的填充区域，insets 和 skinOutline 用于存储该部件轮廓组的内外墙和表面。

可连轮廓组结构体的定义如图 4 所示，其中 IntPoint 结构代表点的坐标，Path 代表一条路径或一个轮廓，Polygons 代表一个可连轮廓组。可连轮廓组是一个二维数组，数组的每一列存储一个轮廓或一条路径。

```
class SliceLayerPart
{
    Polygons outline;
    vector<Polygons> insets;
    Polygons skinOutline;
};
```

图 3　层部件结构体的定义

Fig. 3　Definition of LayerPart

```
struct IntPoint {
    cInt X;
    cInt Y;
}
vector< IntPoint > Path;
vector< Path > Polygons;
```

图 4　可连轮廓组结构体的定义

Fig. 4　Definition of Polygons

### 2.2　层片分组算法

由于使用层间信息继承的路径填充算法的前提是两层层片的轮廓是相似的或相等的，因此需要提前将模型分层后得到的层片进行分组，如图 5 所示。本文提出一种层片分组的算法，能够将模型的层片进行识别和分组。

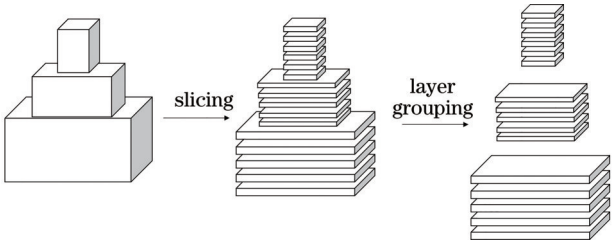


图 5　层片分组

Fig. 5　Layer grouping

层片轮廓相似/相等的定义如下：每一个分组内任意两个层片轮廓的端点个数及线段个数都是相等的，轮廓对应线段的斜率也都是相等的，对应线段的长度成比例或相等。所以本算法需要计算出每个层片轮廓的每条线段的长度及斜率并分别存储到一个数组中。同时使用一个数组存储最后的分组结果，设定第一层为第一个分组的起始层。算法主要步骤如下。

首先，建立轮廓数量、线段长度、线段斜率、分组结果四个二维数组，分别用于存储层片的轮廓数量、轮廓线段的长度、斜率和最后的分组结果。轮廓数量、线段长度、线段斜率这三个数组的每一列

分别存储一个层片的轮廓数量、轮廓线段的长度、斜率。分组结果数组的每一列存储一个分体的所有层片的序号。

然后，按顺序从模型分层完成后生成的层片数据中读取层片的轮廓数据，并将该层片的轮廓数量存入轮廓数量数组中。读取每条线段两个端点的坐标，计算该层片所有轮廓线段的长度和斜率，并按顺序分别存入线段长度和线段斜率数组中。

最后，从第二层开始，比较当前层与前一层的轮廓数量，判断是否相等。若不相等，则新建一个分组，将该层作为该分体的初始层，该层标记为0。若相等，则比较判断这两层对应的轮廓是否相似或相等，若相似或相等则归为同一个分体内，相等标记为1，收缩标记为2，膨胀标记为3；若不相似，则新建一个分组，将该层作为该分体的初始层，并标记0，完成后对下一层重复该步骤，直至完成所有层片的分组操作。流程图如图6所示。

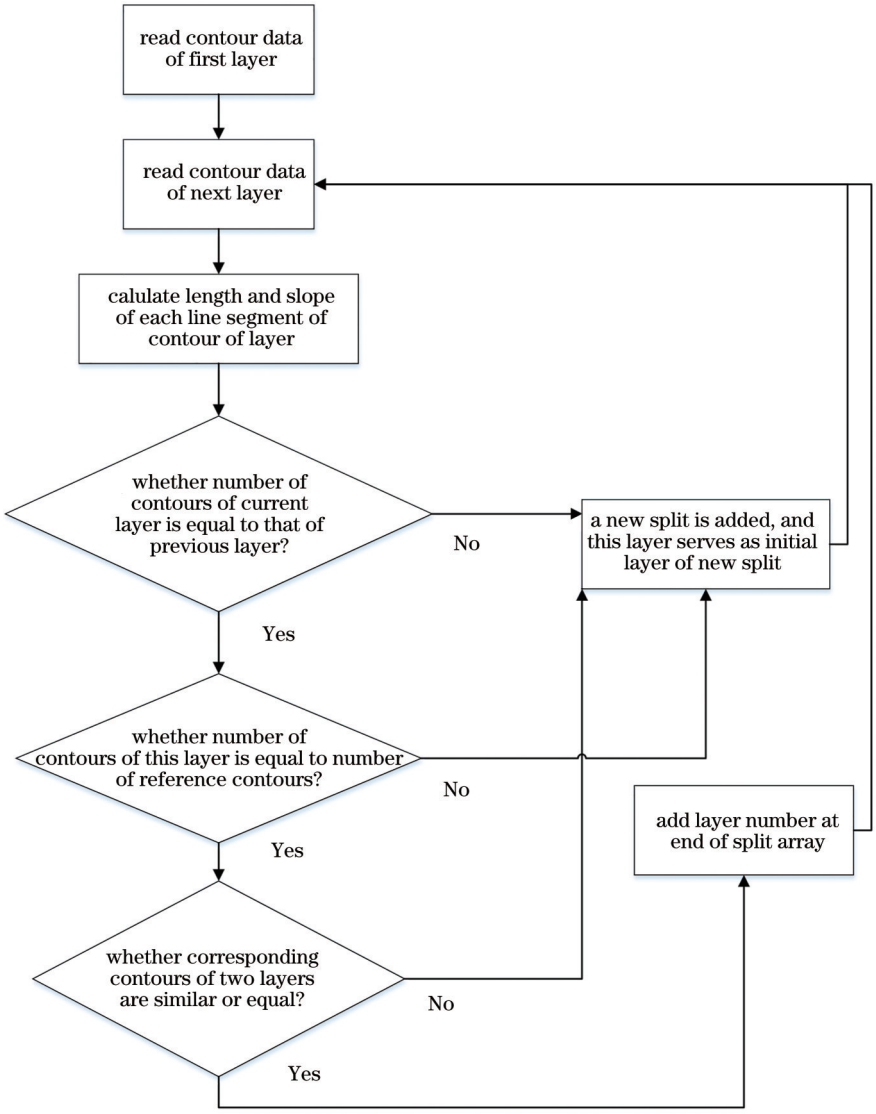


图6　层片分组流程

Fig. 6　Flow chart of layer grouping

### 2.3　基于信息继承的填充算法

使用层片分组算法得到已分组层片集合后，就可以进行填充操作了。

本算法考虑通用的情况，即一层有若干个轮廓。假设下面的前一层填充路径已经用原扫描线填充算法生成了，由于在一个分组内任意两个层片的轮廓可以看作是相似的或相等的，因此后续层片的填充路径可以利用下面一层已生成的填充路径。

由于已完成填充的轮廓和待填充轮廓是相似的，因此两个轮廓编号相同的线段是互相平行的，

我们就可以作一个平行四边形，将已完成填充轮廓对应线段上填充路径的端点加上偏移量，通过“偏移”的操作生成待填充轮廓对应线段上填充路径的端点。已完成填充的轮廓相对于待填充轮廓可能是膨胀、收缩或相等的。所以需要对这三种不同的情况进行处理。

### 2.3.1　相　　等

将已完成填充的轮廓的全部填充路径的端点加上 $X$，$Y$ 方向的偏移量，即可生成待填充轮廓的填充路径，如图 7 所示。

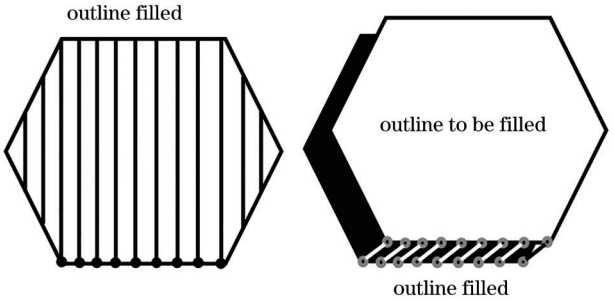


图 7　“相等”示意图

Fig. 7　Schematic of “equalization”

### 2.3.2　收　　缩

只需使用已完成填充轮廓的路径中对应的部分路径，加上 $X$，$Y$ 方向的偏移量，即可生成待填充轮廓的填充路径，如图 8 所示。

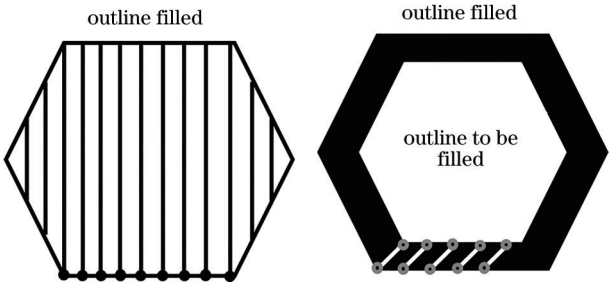


图 8　“收缩”示意图

Fig. 8　Schematic of “shrinking”

### 2.3.3　膨　　胀

利用已完成填充的轮廓的全部路径，此时还需要计算多出来的线段的端点。从已生成的端点中取出相邻的两个端点，分别计算这两个端点在 $X$，$Y$ 方向上的差值。将已生成的端点中的最后一个端点的 $X$，$Y$ 坐标加上差值，计算出后一个端点，以此类推计算出后续端点，完成所有路径的建立，如图 9 所示。

使用图 10 所示的数据结构来临时存储生成的填充线，使用图 11 所示的存储结构来最终存储计算

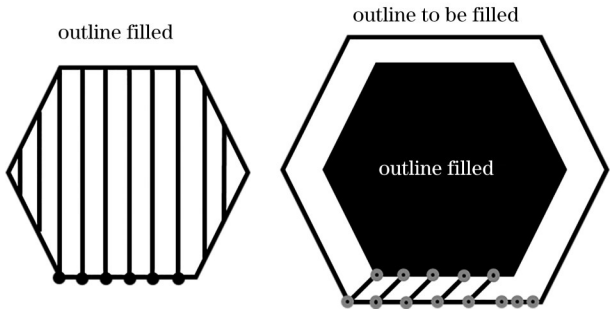


图 9　“膨胀”示意图

Fig. 9　Schematic of “expansion”

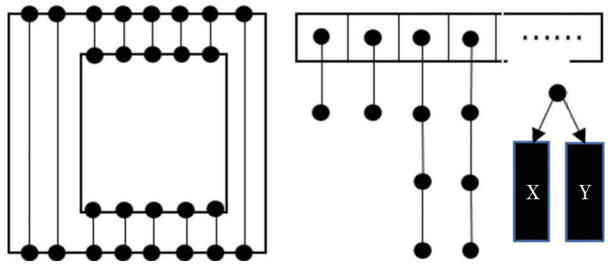


图 10　临时路径存储结构

Fig. 10　Temporary path storage structure

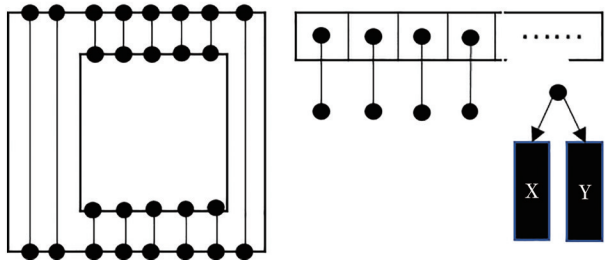


图 11　最终路径存储结构

Fig. 11　Final path storage structure

出的填充线，二维数组的每一列存储一条填充线，以方便后续处理。

具体算法步骤如下。

读取当前层片标记，若当前层标记为 0，则使用扫描线填充算法进行填充；若当前层标记为 1，代表当前层轮廓与前一层轮廓相等，使用上述相等情况下的方法进行填充操作；若当前层标记为 2，代表当前层轮廓比前一层轮廓小，使用上述收缩情况下的方法进行填充操作；若当前层标记为 3，代表当前层轮廓比前一层轮廓大，使用上述膨胀情况下的方法进行填充操作。

由于算法是先处理外轮廓，再处理内轮廓，在生成路径端点的过程中，还需对数组中每列的端点按照 $Y$ 坐标的值从大到小进行排序，以方便最终路

径的生成。

在路径生成完毕后，若当前层轮廓数量大于1，还需要对存储结构进行重构。首先复制存储路径的数组，原数组在后一层路径生成过程中需要被用到。将数组中成员数大于2的列进行拆分，多余的部分添加到数组末尾，确保每一列端点数为2，即每一列存储一条路径，如图11所示。具体算法流程如图12所示，其中result_pre代表已完成填充轮廓的路径集合，result代表待填充轮廓的路径集合，outline_pre代表已完成填充的轮廓集合，outline代表待填充轮廓集合，diff代表轮廓对应线段的偏移量集合，i代表当前轮廓组中轮廓编号索引且初始为0，j代表当前轮廓中线段编号索引且初始为0，poly为一个轮廓所有线段的集合，compute_diff函数用于计算两个轮廓之间对应线段的偏移量，generateLineInfill为标准填充算法，compute为本填充算法函数，scanlineidx0和scanlineidx0_2为起始扫描线，scanlineidx0和scanlineidx0_2为末尾扫描线，gap和gap_2为扫描线的数量。

```
Definition
1. diff=compute_diff(outline_pre, outline);
2. If flag=0
3. generateLineInfill(outline);
4. return;
5. Repeat    i ++
6. outline_pre[i];
7. outline[i];
8.     j =0
9. Repeat    j ++
10.If flag=1
11.result = compute(result_pre, diff);
12.scanlineidx0=ScanIdx(poly[j],line_distance) + 1;
13.scanlineidx1=ScanIdx(poly[j], line_distance);
14.gap = scanlineidx1- scanlineidx0;
15.scanlineidx0_2=ScanIdx(poly[j], line__ distance) + 1;
16.scanlineidx1_2=ScanIdx(poly[j],line_distance);
17.gap_2 = scanlineidx1_2- scanlineidx0_2;
18.If flag=2
19.result_temp = result_pre[gap_2-gap];
20.result = compute(result_temp, diff);
21.If flag=3
22.result = compute(result_pre, diff);
23.k=0;
24.Repeat    k ++
25.result+= (temp_point, diff);
26.until k >gap-gap2;
27.until j>poly.size()
28.until    i>polygon.size()
29.End    Algorithm 1
```

图12　基于层间信息继承的填充算法

Fig. 12　Filling algorithm based on information inheritance between layers

可以看出，本算法不需要循环求扫描线集合与轮廓线的交点，也不需要将扫描线集合与轮廓线的交点依次连接得到路径。直接利用已生成的路径信息，生成本层的路径，大大减少了算法的计算量。

## 3　实验结果

### 3.1　实验数据与环境

实验环境为：操作系统为Ubuntu18.04，处理器型号为Intel Core i9-8750H，内存为64 G，编程语言为C++，GCC版本为7.5，编程平台为Visual Code。本次实验所用的STL文件来源于SolidWorks软件建模。

### 3.2　算法正确性验证

本文使用SolidWorks软件自主建模的测试模型，进行算法结果正确性的验证。测试模型如图13所示。图14(a)、(b)、(c)分别为该模型第

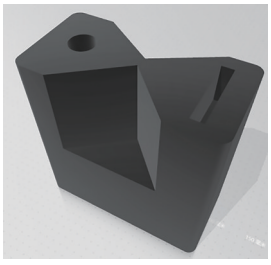

图13　测试模型

Fig. 13　Test model

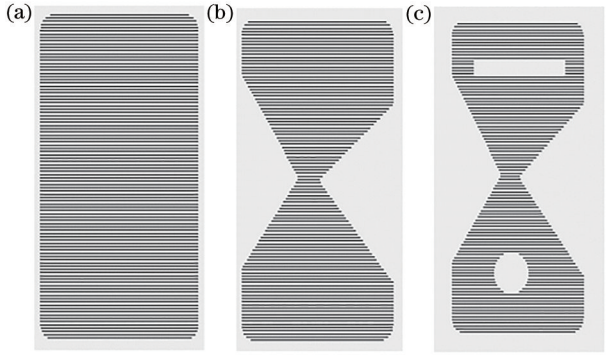


图 14　轮廓填充结果。(a) 第 100 层；(b) 第 1500 层；(c) 第 1800 层

Fig. 14　Contour filling results. (a) 100th floor; (b) 1500th floor; (c) 1800th floor

100 层、第 1500 层、第 1800 层使用本算法填充后的图形，可以看出，本算法能够正确地计算出填充路径。

### 3.3　实验对比

实验首先比较了 CuraEngine 切片软件（开源切片引擎）中扫描线填充算法（算法 1）与本文算法（算法 2）。在相同切片厚度的情况下，用两种路径填充算法对不同模型进行路径填充处理。目前的层片分组算法对形状规则轮廓的分组效果比较好，对复杂轮廓的分组效果比较差，会降低本文填充算法的效率，故选取了较多轮廓较为规整的模型进行测试。路径填充部分的性能比较结果如表 1 所示。

表 1　不同填充算法对不同模型的填充耗时

Table 1　Time-consumption comparison of different filling algorithms for different models

| Model | Size /(mm×mm) | Time-consumption of algorithm 1 /s | Time-consumption of algorithm 2 /s | Acceleration ratio |
|---|---|---|---|---|
| Cube | 300×300 | 60.061 | 24.092 | 2.49 |
| Trapezoidal platform | 450×450 | 90.092 | 56.948 | 1.58 |
| Handstand trapezoid platform | 450×450 | 91.060 | 59.323 | 1.53 |
| Curved cube | 133×100 | 22.807 | 14.802 | 1.54 |
| Box | 300×300 | 79.716 | 36.558 | 2.18 |
| Test model | 200×100 | 25.576 | 15.470 | 1.65 |
| Vase | 145×145 | 91.593 | 90.108 | 1.02 |
| Sculpture | 143×143 | 73.962 | 70.44 | 1.05 |

由表 1 可以看出，对于几何规整的模型，本文所提算法与 CuraEngine 中的算法相比，在填充耗时上降低了很多。其中，在等截面的情况下，本文算法的效果最好，耗时相比原算法减少了 60% 左右；对于变截面的情况，本文算法相比于原算法，耗时减少了 35% 左右；对于其他情况，本文算法也有较好效果，但对于表面较为复杂的模型，本文算法没有得到较好的效果，如花瓶模型，这是由于模型表面较为复杂，层片分组算法没有较好地进行分组，层片分组的个数较多。

我们使用上述自主建模的测试模型，进行不同扫描线间距和不同层数的测试，算法耗时对比分别如图 15 和图 16 所示。

由图 15 可以看出，本文算法的耗时随扫描线间距的增加而增加。扫描线间距越小，本文算法的加速效果就越好。这是由于扫描线间距越小，每一层的路径就越多，原算法求交的次数就越多，而本文算法不需进行求交，直接利用已有路径来生成路径，因此与轮廓相交的扫描线数量越多，本文算法的加速效果就越好。

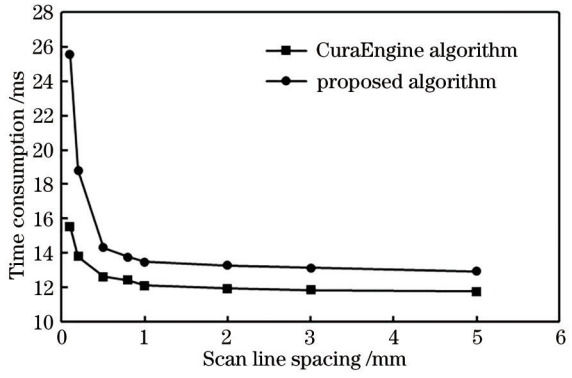


图 15　不同算法耗时随扫描线间距的变化曲线

Fig. 15　Time consumption of each algorithm versus scan line spacing

由图 16 可以看出，随着模型层数的增加，本文算法的耗时呈线性递增趋势，模型的层数对算法效率没有较大的影响。

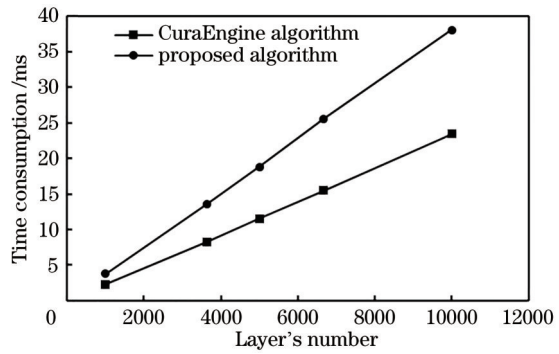


图 16　不同算法耗时随模型层数的变化曲线

Fig. 16　Time consumption of each algorithm versus layer number

## 4　结　　论

针对大数据量模型路径规划所需时间陡增的问题，提出了基于层间信息继承的填充算法。根据模型的几何特征，首先对模型分层后的层片进行分组操作，在每个分组内利用任意两层间的信息继承特性，快速计算出当前层的填充路径，加快算法的执行速度。在对规则模型进行路径填充时，所提算法可以得到较好的加速效果，并且扫面线的间距越小，所提算法的加速效果就越明显。研究结果对实际的工业制造具有重大意义。

## 参　考　文　献

[1] Lu B H. Additive manufacturing: current situation and future[J]. China Mechanical Engineering, 2020, 31(1): 19-23.
卢秉恒. 增材制造技术: 现状与未来[J]. 中国机械工程, 2020, 31(1): 19-23.

[2] Wang Q H, Sun A L. Application and development of 3D printing technology in composite materials manufacturing[J]. Fiber Reinforced Plastics, 2015(4): 9-14.
王强华, 孙阿良. 3D 打印技术在复合材料制造中的应用和发展[J]. 玻璃钢, 2015(4): 9-14.

[3] Zhang X J, Tang S Y, Zhao H Y, et al. Research status and key technologies of 3D printing[J]. Journal of Materials Engineering, 2016, 44(2): 122-128.
张学军, 唐思熠, 肇恒跃, 等. 3D打印技术研究现状和关键技术[J]. 材料工程, 2016, 44(2): 122-128.

[4] Liu J, Lei Z J, Gu H Q, et al. Domestic developing status of 3D pringting in China[J]. Manufacturing Technology & Machine Tool, 2015(3): 17-21, 25.
柳建, 雷争军, 顾海清, 等. 3D打印行业国内发展现状[J]. 制造技术与机床, 2015(3): 17-21, 25.

[5] Lu B H, Li D C. Development of the additive manufacturing (3D printing) technology[J]. Machine Building & Automation, 2013, 42(4): 1-4.
卢秉恒, 李涤尘. 增材制造(3D打印)技术发展[J]. 机械制造与自动化, 2013, 42(4): 1-4.

[6] Yang Q, Lu Z L, Huang F X, et al. Research on status and development trend of laser additive manufacturing[J]. Aeronautical Manufacturing Technology, 2016, 59(12): 26-31.
杨强, 鲁中良, 黄福享, 等. 激光增材制造技术的研究现状及发展趋势[J]. 航空制造技术, 2016, 59(12): 26-31.

[7] Ding D H, Pan Z X, Cuiuri D, et al. A tool-path generation strategy for wire and arc additive manufacturing[J]. The International Journal of Advanced Manufacturing Technology, 2014, 73(1/2/3/4): 173-183.

[8] Ren F, Sun Y W, Guo D M. Combined reparameterization-based spiral toolpath generation for five-axis sculptured surface machining[J]. The International Journal of Advanced Manufacturing Technology, 2009, 40(7/8): 760-768.

[9] He H Q, Yang P. Global subdivision algorithm on convex decomposition of concave polygon[J]. Journal of Civil Aviation University of China, 2011, 29(3): 52-55.
贺怀清, 杨鹏. 一种凹多边形凸分解的全局剖分算法[J]. 中国民航大学学报, 2011, 29(3): 52-55.

[10] Griffiths J G. Toolpath based on Hilbert's curve[J]. Computer-Aided Design, 1994, 26(11): 839-844.

[11] Gan Q. General scan-line polygon-filling algorithm [J]. Computer Engineering and Applications, 2000, 36(2): 57-59.
甘泉. 通用扫描线多边形填充算法[J]. 计算机工程与应用, 2000, 36(2): 57-59.

[12] Aiyiti W, Xiang L, Zhang L Z, et al. Study on the veritable parameters filling method of plasma arc welding based rapid prototyping[J]. Key Engineering Materials, 2012, 522: 110-116.

[13] Huang X M, Niu Z W, Dong X J. An algorithm of sub-regional scanning path generation for rapid prototyping manufacturing[J]. Machine Design & Research, 2007, 23(1): 80-82.
黄雪梅, 牛宗伟, 董小娟. 快速成型技术中的分区扫描路径产生算法[J]. 机械设计与研究, 2007, 23(1): 80-82.

[14] Jin G Q, Li W D, Gao L, et al. A hybrid and adaptive tool-path generation approach of rapid

prototyping and manufacturing for biomedical models [J]. Computers in Industry, 2013, 64(3): 336-349.

[15] Ma L, Huang W D. An algorithm for rapid recognition of outlines group based on information inheriting[J]. Mechanical Science and Technology for Aerospace Engineering, 2009, 28(4): 482-486.
马良, 黄卫东. 基于信息继承的可连轮廓组快速提取算法[J]. 机械科学与技术, 2009, 28(4): 482-486.