



山东大学
SHANDONG UNIVERSITY

毕业论文(设计)

论文(设计)题目:

连通费马螺旋线路径的交互式生成系统

姓 名 惠浩
学 号 201905130190
学 院 计算机科学与技术
专 业 人工智能
年 级 2019
指导教师 赵海森

2023 年 05 月 20 日

摘 要

连通货马螺旋线是一种空间填充曲线，它能填充任何拓扑连通的区域，同时满足全局连续性和平滑性，已经应用于 FDM 打印，数控加工，陶瓷打印，机器人轨迹规划等多个领域。而现有的生成连通货马螺旋线算法实现程序并不能很好地满足用户的某些需求。其原因如下：一、程序的输入是由离散的点构成的图形轮廓。用户要首先在专业的二维绘图软件中绘制出图形，然后再通过编程其转换成离散的点，使用门槛较高；二、程序的输出是一系列组成连通货马螺旋线的点坐标，存储在纯文本文件中，并不直观，缺少可视化功能。为了解决用户在生成连通货马螺旋线过程中的难点，本文基于已有的生成连通货马螺旋线算法代码开发了连通货马螺旋线的交互式生成系统，使得生成连通货马螺旋线更简单，降低了其使用门槛，提升了用户体验。本文的工作主要有以下两点：一、本文开发了一个简单易用的路径编辑器，用来编辑算法的输入图形，在实现了核心功能的前提下，比专业矢量图形编辑软件更简单易用；二、本文开发了一个直观的连通货马螺旋线可视化系统，有动画、仿真、三维三种可视化方式，满足了用户不同的可视化需求。

关键词：连通货马螺旋线；可视化；人机交互；路径编辑；立即模式 GUI

ABSTRACT

The connected Fermat spirals are space-filling curves that can fill any topologically connected region while satisfying global continuity and smoothness, and have been used in many fields such as FDM printing, CNC machining, ceramic printing, and robot trajectory planning. However, the existing implementation of the algorithm for generating connected Fermat spirals is not well suited to meet some of the user's needs. The reasons for this are as follows: First, the input to the program is a graphical outline composed of discrete points. The user has to first draw the graph in professional 2D drawing software, and then convert it into discrete points by programming, which is a high threshold; second, the output of the program is a series of point coordinates that make up the connected Fermat Spiral, stored in a plain text file, which is not intuitive and lacks visualization capabilities. In order to solve the difficulties of users in the process of generating connected Fermat spirals, this paper develops an interactive generation system for connected Fermat spirals based on the existing code of the algorithm for generating connected Fermat spirals, which makes it simpler to generate connected Fermat spirals, reduces the threshold of its use, and improves the user experience. In this paper, we have developed an easy-to-use path editor to edit the input graphics of the algorithm, which is simpler and easier to use than professional vector graphics editing software while achieving the core functions; second, we have developed an intuitive visualization system for the connected Fermat spiral, with three visualization methods: animation, simulation, and 3D, to meet different visualization needs of users.

Keywords: connected Fermat spirals, visualization, HCI, path editor, IMGUI

目 录

| | | |
|-------|--------------|----|
| 第 1 章 | 绪论 | 1 |
| 1.1 | 选题背景和研究意义 | 1 |
| 1.2 | IMGUI | 2 |
| 1.3 | SVG | 3 |
| 1.4 | 论文组织结构 | 3 |
| 第 2 章 | 需求分析 | 5 |
| 2.1 | 功能性需求 | 5 |
| 2.1.1 | 路径编辑器 | 6 |
| 2.1.2 | 连通费马螺旋线路径可视化 | 8 |
| 2.2 | 非功能性需求 | 9 |
| 第 3 章 | 路径编辑器 | 10 |
| 3.1 | 路径编辑器的交互设计 | 11 |
| 3.1.1 | 市场调研 | 11 |
| 3.1.2 | 设计目标 | 12 |
| 3.1.3 | 通用交互方式 | 12 |
| 3.1.4 | 核心交互设计 | 13 |
| 3.1.5 | 其他功能交互设计 | 14 |
| 3.2 | 路径编辑器的界面设计 | 17 |
| 3.2.1 | 布局 | 17 |
| 3.2.2 | 颜色 | 18 |
| 3.2.3 | 图标 | 18 |
| 3.3 | 路径编辑器的具体实现 | 19 |
| 3.3.1 | 悬停 | 19 |
| 3.3.2 | 选中 | 22 |
| 3.3.3 | 平移和缩放 | 22 |
| 3.3.4 | 状态机 | 23 |
| 3.3.5 | 导出、导入路径 | 23 |
| 第 4 章 | 连通费马螺旋线的可视化 | 24 |

| | | |
|-----------|-----------------------|----|
| 4.1 | 连通费马螺旋线可视化的交互设计 | 24 |
| 4.1.1 | 可视化类型：动画 | 25 |
| 4.1.2 | 可视化类型：仿真 | 27 |
| 4.1.3 | 可视化类型：三维 | 27 |
| 4.2 | 连通费马螺旋线可视化的界面设计 | 27 |
| 4.2.1 | 布局 | 27 |
| 4.2.2 | 颜色 | 27 |
| 4.3 | 连通费马螺旋线可视化的具体实现 | 28 |
| 4.3.1 | 拉普拉斯平滑 | 28 |
| 4.3.2 | 路径采样 | 28 |
| 4.3.3 | 平移和缩放 | 29 |
| 4.3.4 | 路径动画化 | 29 |
| 4.3.5 | 欠填充和过填充可视化 | 30 |
| 4.3.6 | 三维可视化 | 31 |
| 第 5 章 | 结束语 | 32 |
| 5.1 | 本文总结 | 32 |
| 5.2 | 工作展望 | 32 |
| 5.2.1 | 探索自动从图片中获取路径 | 32 |
| 5.2.2 | 探索真实的物理模型 | 32 |
| 参考文献 | | 33 |
| 致 谢 | | 35 |
| 附录 1 译文中文 | | 36 |
| 附录 2 译文原文 | | 48 |

第 1 章 绪论

1.1 选题背景和研究意义

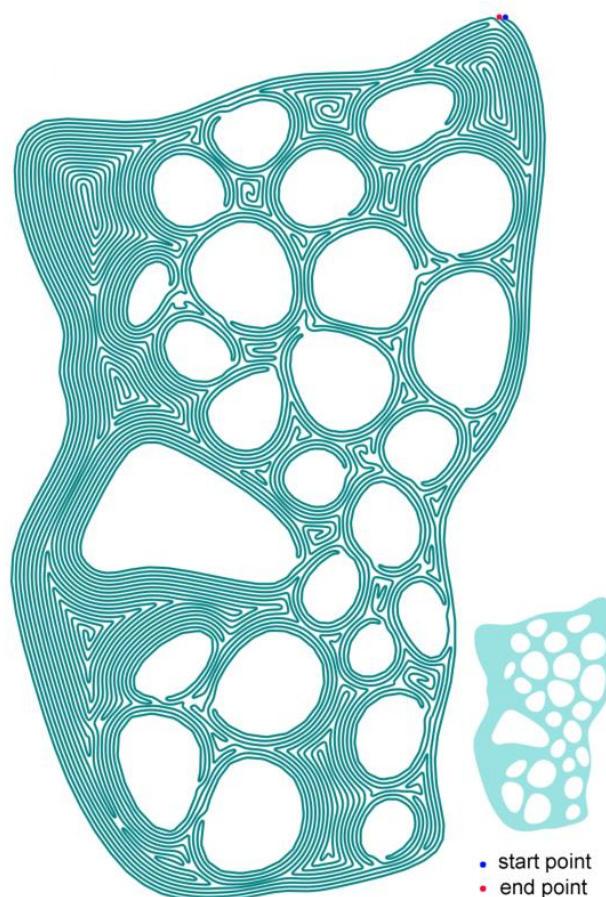


图 1-1 连通货马螺旋线示例，右下角是输入的图形^[1]

如图 1-1 所示，连通货马螺旋线是一种新的应用广泛的空间填充曲线。与传统的空间填充曲线不同，连通货马螺旋线对任意拓扑连通的区域都可以生成一条全局连续且平滑的空间填充曲线^[2]。连通货马螺旋线已经应用于 FDM 打印，数控加工，陶瓷打印和机器人轨迹规划等多个领域。尽管已经有了生成连通货马螺旋线路径的算法实现程序，但是并不能满足某些用户的需求，主要有以下两个原因：一、程序的输入是由离散的点构成的图形轮廓。用户要首先在专业的二维绘图软件中绘制出图形，然后再通过编程其转换成离散的点，使用门槛较高；二、程序的输出是一系列组成连通货马螺旋线的点坐标，存储在纯文本文件中，并不直观，缺少可视化功能。

为了解决这个问题，本文基于已有的生成连通货马螺旋线路径算法，使用 ImGui（Immediate Mode Graphic User Interface）范式设计了一个简单而高效的连通货马螺旋线路径的交互式生成系统，旨在满足学术界和工业界的相关需求。通过这个系统，用户可以轻松地创建和编辑生成连通货马螺旋线算法的输入二维图形，同时，由于其采用 SVG（Scalable Vector Graphics）格式作为算法输入的二维图形存储格式，用户可以很轻易地导入外部（网上下载或从专业矢量图编辑软件中导出）的 SVG 图形。用户可以很容易看到生成的连通货马螺旋线的可视化，并对其做出调整。连通货马螺旋线路径的交互式生成系统将降低使用连通货马螺旋线的使用门槛，提升用户使用体验和生成连通货马螺旋线效率，促进连通货马螺旋线的应用。下面将介绍本文的一些相关工作，如 ImGui、SVG 等。

1.2 ImGui

像 QT^[3]、wxWidgets^[4]或 tkinter^[5]等当下最先进的跨平台图形用户界面（GUI）库所做的那样，将部件与对象或者句柄绑定的方法被称为 RMGUI（Retained Mode GUI），与之相对的方法是 ImGui（Immediate Mode GUI）^[6]。

ImGui 最早由 Casey Muratori 在 2015 年提出。因为 ImGui 不保存 UI 状态，所以即使 UI 没有发生变化，也要进行重新绘制。这样做的好处是，因为不保存状态，也就没有了状态之间的同步，不保存状态也无需在状态改变时双向拷贝数据，所以 ImGui 能够大大简化开发的难度。但是对于当时的电脑性能来说每一帧都要重新绘制占用了大量的 CPU 资源，所以在当时并没有流行起来，但是对于今天的 CPU 和 GPU 来说，一秒 60 帧的绘制是非常轻松和容易的。所以 Sean Barret 将 RMGUI 描述为对更简单的 ImGui 系统的“过早优化”，因为引入了部件的生命周期管理和额外的心智负担^[7]。

由于 ImGui 相比 RMGUI 的众多好处，本文采用了 ImGui 范式作为 GUI 实现的范式，本文并没有从头实现一个 ImGui 的用户界面库，而是选用 ImGui 的一种流行的实现：Dear ImGui（<https://github.com/ocornut/imgui>）。它有以下四个目标^[8]：

1. 最小化状态同步；
2. 最小化用户端状态存储；
3. 最小化设置和维护；

4. 易于创建动态 UI。

尽管现在 ImGui 还没有一个公认的定义，各种 ImGui 的实现也都有细节上的差别，但是 Dear ImGui 还是给出了自己的定义^[8]：

1. ImGui 是应用程序和 UI 系统之间的接口，是一个视图将应用程序必须保留的于 UI 系统有关的数据降到最低的接口，是一个视图最大限度地减少 UI 系统必须保留与应用程序相关数据的接口。
2. ImGui 不设计到接口的具体实现。不依赖于任何 UI 系统。

1.3 SVG

SVG，即可缩放矢量图形（Scalable Vector Graphics）是一种用 XML 来描述二维图形的语言^[9]。

SVG 相比于点阵存储的图形，具有缩放不失真，存储占用小等优点，已经得到了广泛的应用。比如，像 Inkscape, Adobe Illustrator 等当下最先进的矢量图编辑软件均可导出为 SVG 文件格式。

出于 SVG 的多种优点和于其他矢量图编辑软件互通的原因，本文采用了 SVG 格式作为生成连通货马螺旋线算法输入的二维图形存储格式。

1.4 论文组织结构

本文以连通货马螺旋线路径的使用背景展开，讲述了连通货马螺旋线路径的交互式生成系统的需求、设计和实现。最后分析了本系统的一些未来可以改进的地方。以下是本文各章节的安排：

第一章为绪论部分，简要介绍了连通货马螺旋线路径，其优点和广泛应用，以及为什么要开发一个连通货马螺旋线路径的交互式生成系统。除此之外还介绍了于连通货马螺旋线路径的交互式生成系统相关的技术、相关工作。并在本章的最后介绍了本文的组织结构

第二章介绍了要开发一个连通货马螺旋线路径交互式生成系统都有哪些需求，进行了功能性需求和非功能性需求的分析。在功能性需求分析中得出开发一个连通货马螺旋线路径交互式生成系统的两个子系统：路径编辑器系统和可视化系统，分别对应这连通货马螺旋线路径交互式生成系统的输入和输出。

第三章详细地介绍了路径编辑器子系统，包括它的交互设计、界面设计以及具

体实现。

第四章详细地介绍了可视化子系统，包括它的交互设计、界面设计以及具体实现。

第五章为结束语部分，对本文基于现有连通货马螺旋线算法的连通货马螺旋线路径交互式生成系统的设计 and 实现工作进行总结，以及未来工作的展望。

第 2 章 需求分析

一个软件系统的需求分析是分析一个软件需要完成哪些功能——即这个系统需要“实现什么”，而不是分析这个系统“怎么实现”^[10]。本节将从功能性需求、非功能性需求以及设计约束三个方面对连通货马螺旋线路径的交互式生成系统进行需求分析。

2.1 功能性需求

功能性需求即软件必须完成哪些事情、哪些功能，是最主要的需求^[11]。

如图 2-1 所示现有生成连通货马螺旋路径算法程序运行的流程，用户首先生成用点表示的拓扑连通的区域，然后程序读取，运行后输出生成的连通货马螺旋线路径，这个连通货马螺旋线路径也是用离散的点所表示的。

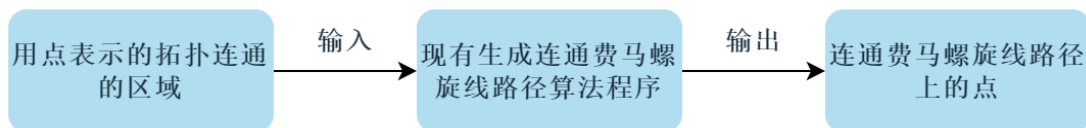


图 2-1 现有生成连通货马螺旋线路径算法程序运行流程

而现有的难点在于用户如何获得用点表示的拓扑连通区域，以及用户在获取连通货马螺旋线路径上的点之后需要手动编程来实现可视化。通过用户调研，我们发现大多数用户使用图 2-2 所示的流程来生成用点表示的拓扑连通区域。用户首先会用一些专业的矢量图编辑软件（如 Inkscape，Adobe Illustrator 等）创建并编辑一个拓扑连通区域的图形，然后将其导出为某种矢量图格式（一般为 SVG），最后通过编程的方式将图形转换为离散的点保存在文件中作为生成连通货马螺旋线路径算法程序的输入。这就要求使用生成连通货马螺旋线路径算法程序的用户需要会使用专业的矢量图编辑软件，以及能够通过编程将一个二维矢量图转换成为离散的点。



图 2-2 生成用点表示的拓扑连通的区域

总而言之，对现有生成连通货马螺旋线路径算法程序的用户而言，至少需要有

以下能力：

1. 能够使用专业的矢量图编辑软件创建并编辑图形
2. 能够通过编程将二维矢量图形转换为离散的点
3. 能够通过编程对生成的用点表示的连通货马螺旋线路径进行可视化。

这三个要求给使用连通货马螺旋线路径设置了门槛，提升了其使用难度。且就算用户这三个要求全部满足，由于这些功能并不都是集成在一个软件里面，不同软件之间来回切换相对繁琐，增加了使用上的割裂感，没有很好的集成度。

根据以上对用户使用现有生成连通货马螺旋线路径算法程序过程的分析，如表 2-1 所示，本文总结出了用户在使用过程中遇到的难点以及对应的需求。

表 2-1 使用难点和对应需求

| 使用难点 | 对应需求 |
|---|-------------------------------|
| 专业矢量图编辑软件需要一定的学习和上手门槛 | 需要一个简单、易上手的图形编辑器 |
| 需要通过编程将二维矢量图形转换为离散的点 | 需要将二维矢量图形转换为离散的点的功能 |
| 需要通过编程将连通货马螺旋线路径进行可视化 | 需要连通货马螺旋线路径可视化的功能 |
| 矢量图形编辑器、转换成离散的点的程序、生成连通货马螺旋线的程序、可视化程序之间来回切换过于繁琐 | 需要将这几个功能全部集成在一个软件里面，减少使用中的割裂感 |

所以，一个连通货马螺旋线路径的交互式生成系统的功能性需求大体上有路径编辑器、连通货马螺旋线路径可视化、路径采样（将路径转换为离散的点）。将其命名为路径编辑而不是图形编辑器的是因为本文认为一个拓扑联通的区域可以看作是闭合的路径。为了简化系统的模型，本文将路径采样作为可视化的一个子功能。下面，将会对路径编辑器和连通货马螺旋线路径可视化分别进行功能性需求分析。

2.1.1 路径编辑器

作为一个路径编辑器，其核心功能必须是能够对路径进行编辑，包括新建路径、编辑路径形状、移动路径、删除路径、缩放路径。其中缩放路径有按照宽度进行缩

放、按照高度进行缩放、等比缩放三种缩放方式。因为要生成连通货马螺旋线路径的区域一般都是拓扑连通的区域，所以需要有闭合路径这个功能来确保是一个拓扑联通的区域。

由于用户在没有任何参照的情况下去编辑一个已有中的二维图形是有一定难度的，所以路径编辑器需要有根据图片编辑路径这个功能。这样用户在看到一个满意的二维图形后就可以将其图像插入到路径编辑器中，然后以图片作为参考进行编辑。所以这就要求路径编辑器有简单的编辑图像的功能：插入图像、移动图像、删除图像、缩放图像、删除图像、上移/下移图像、锁定/解锁图像。因为不同来源的图像尺寸不同，所以提供了缩放图像的功能，和缩放路基相似，也有按照宽度缩放、按照高度缩放、等比缩放三种缩放方式。在插入多张图像时，就有了图像之间的遮挡关系，所以需要有上移/下移图像的功能。因为图像只是作为编辑路径的一个参照，在调整好图像之后要减少其对路径编辑的影响，所以需要有锁定图像的功能，在图像锁定之后，将无法再对图像进行编辑直到对图像进行解锁操作。

路径编辑器作为图形编辑器的一种，当然也需要有图形编辑器的一些基础功能：移动画布、缩放画布、撤销、重做、复制、剪切、粘贴、导入路径、导出路径。因为路径是保存在内存中的三次贝塞尔曲线的节点，用户在退出软件之后，所编辑的路径就会丢失，所以需要有导出的功能来保存用户编辑的路径到存储中。同时导出的路径格式要尽可能能够与其他专业矢量图编辑软件所互通，比如有的用户能够熟练使用专业的矢量图编辑软件，所以路径编辑器导入路径需要能够导入其他专业矢量图编辑软件导出的特定格式的路径，同时路径编辑器导出的路径能够被其他专业的矢量图编辑软件所导入。这样增加了本软件与其他软件之间的互通性，为用户提供了更大的便利。

虽然路径编辑器是一个编辑器，但是也提供了预览模式这个功能，这样能够看到编辑好的路径的全貌，减去编辑相关的视觉干扰。

表 2-2 总结了路径编辑器要实现的功能，以及每个功能需要的子功能。

表 2-2 路径编辑器的功能

| 功能 | 子功能 |
|-----------------------|-----------------|
| 路径编辑（路径编辑器的核心功能） | 新建路径 |
| | 编辑路径形状 |
| | 缩放路径（宽、高、等比） |
| | 移动路径 |
| | 删除路径 |
| | 闭合路径 |
| 简单的图像编辑（以图像为参考进行路径编辑） | 插入图像 |
| | 缩放图像（宽、高、等比） |
| | 删除图像 |
| | 移动图像 |
| | 上移/下移图像 |
| | 锁定/解锁图像 |
| 编辑器的基础功能 | 移动、缩放画布 |
| | 撤销、重做 |
| | 复制、剪切、粘贴 |
| | 导出编辑的路径 |
| | 导入本软件或其他软件导出的路径 |
| 其他 | 路径预览 |

2.1.2 连通费马螺旋线路径可视化

因为生成连通费马螺旋线路径算法的输出是由离散的点组成的拓扑连通区域，所以在可视化前需要对输入的闭合路径进行采样，将其转换成一系列离散的点。

由于生成连通费马螺旋线路径算法的输出是路径上一系列的点坐标，对于用户来说并不直观，无法直接看到生成的连通费马螺旋线路径到底是什么样的，所以可视化功能需要能够展示出生成的连通费马螺旋线路径的形状。连通费马螺旋线是一条全局连续的曲线，有唯一的起点和唯一的终点，所以在打印的过程是没有中断的。基于以上特点，本文提供了连通费马螺旋线从起点到终点的动画可视化，模拟了真实的打印过程。

连通货马螺旋线是一种空间填充曲线，它是有宽度的，只可视化它的形状是不够的。生成的连通货马螺旋线有可能无法覆盖一些区域，也有可能重复覆盖一些区域，这些欠填充、过填充的区域是会影响打印质量的，用户需要知道这些区域。所以本系统需要有可视化欠填充、过填充区域的功能。

除了上面两种可视化方式之外，本系统还提供了三维可视化，将连通货马螺旋线以三维的方式展现出来，让用户以不同视角查看生成的连通货马螺旋线路径。

2.2 非功能性需求

非功能性需求主要分为目标系统限制和开发维护限制所带来的需求^[11]。

连通货马螺旋线的用户有学术界的有工业界的，他们所使用的操作系统不同，这要求连通货马螺旋线路径的交互式可视化系统需要有跨系统运行的能力，所以本文在采用了跨平台用户界面框架 **ImGui** 进行开发。

性能需求。用户所输入的用来生成连通货马螺旋线路径的图形可能非常复杂，所以本文使用高性能编程语言 **C++** 来进行开发。

第 3 章 路径编辑器

前面的需求分析章节已经分析了路径编辑器是“做什么的”，需要有“哪些功能”，而本章节将要探讨如何实现前面所提到的功能。

首先就是路径编辑器的核心功能：编辑路径。而在讨论如何实现编辑路径之前还有一个必须解决的问题：如何表示路径？不同的路径表示方法将会决定编辑路径的方式。

首先路径本质上是一条曲线，控制一条曲线最简单的方法就是用有限的点去拟合这条曲线：即给定一系列点然后通过插值的方式得到曲线，但是无法准确控制两个点之间曲线的形状。所以在实际应用中，通常会采取曲线只逼近点的方案。在逼近方案中，控制点影响曲线的形状，但不精确指定曲线的形状^[12]。贝塞尔曲线是图形学中一种重要的逼近曲线，一条贝塞尔曲线是一条逼近其控制节点的多项式曲线。贝塞尔曲线的多项式次数可以是任意的，一个 d 次的贝塞尔曲线由 $d + 1$ 个控制节点所控制。贝塞尔曲线会经过其第一个和最后一个控制节点，其他控制节点则会影响贝塞尔曲线的形状。通常，复杂的图形是由一些低阶的贝塞尔曲线连接所组成，一般是三次贝塞尔曲线^[12]。本文使用多条三次贝塞尔曲线连接组成一条曲线路径，公式 3-1 为三次贝塞尔曲线的表达式。

$$\mathbf{B}(t) = (1 - t)^3 \mathbf{P}_0 + 3(1 - t)^2 t \mathbf{P}_1 + 3(1 - t) t^2 \mathbf{P}_2 + t^3 \mathbf{P}_3, 0 \leq t \leq 1. \quad (3 - 1)$$

三次贝塞尔曲线一共有四个控制节点，曲线路径会经过第一个和最后一个控制节点。为了方便起见，在后文中，路径节点指的是第一个和最后一个控制节点，而控制节点则特指第二个和第三个控制节点。

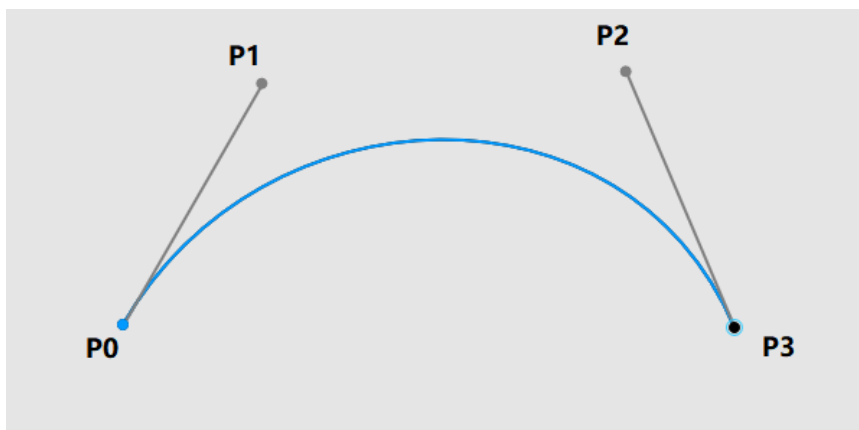


图 3-1 三次贝塞尔曲线

3.1 路径编辑器的交互设计

3.1.1 市场调研

常用的专业矢量图形编辑器都有三次贝塞尔曲线路径编辑器这个功能，本文调研了如 Adobe Illustrator 和 Inkscape 的交互设计，它们的交互设计大体相同，有略微区别。以下是它们在交互设计上相同的地方：

1. 插入节点和移动节点使用不同的工具。
2. 当完成插入节点后，松开鼠标左键后，会立即进入下一个节点的插入。直到闭合路径或者按下指定的退出快捷键为止。
3. 当插入节点完成后，不立即松开鼠标左键，而是继续拖动，可以拖拽出控制节点的前一个和后一个控制节点，这三个控制节点是共线且等距的，这是为了两段三次贝塞尔曲线在连接点处具有平滑性。
4. 如果在插入节点的过程中想对现有路径形状进行编辑，则使用指定快捷键来暂时切换到
5. 移动控制节点总是有共线和等距这两个限制，如果需要去除这两个限制，需要额外的步骤（切换工具，或使用快捷键）。

虽然它们确实都可以满足连通货马螺旋线路径交互式生成系统的需求，但是过于繁琐，以下是造成其繁琐的两个原因：

第一个原因是共线和等距不是必要的需求。上面两个软件为了三次贝塞尔曲线在接口处的平滑性设定了这两个默认的需求，但是连通货马螺旋线路径算法的输入是有离散的点组成的拓扑连通图形，连连续性都没有，更别谈平滑性，其次如果需要平滑性，可以在生成这些点的过程中进行拉普拉斯平滑。

第二个原因是插入节点完成后默认进入新节点的插入。因为插入完成后拖动只能表示前一个控制节点的位置，后一个控制节点的位置和前一个控制节点的位置是基于当前路径节点中心对称的，这就导致除了很规则的图形，基本上很少能调整到正好完美拟合图形，同时由于后一个控制节点的中心对称，导致还会影响下一段三次贝塞尔曲线的拟合；这就不得不导致每插入完一个节点后都要手动切换到移动模式，解除控制节点的共线和等距限制，然后调整控制节点直到完美拟合，步骤非常繁琐。虽然可以一次性插入完所有节点后在统一进行调整，但是在插入的时

候又不太好把控插入节点之间的距离，如果两个节点之间的图形比较复杂，则可能无论怎么调整都不能很好拟合。

3.1.2 设计目标

针对现有解决方案应用在连通货马螺旋线路径交互式生成系统的局限性。本文提出了以下几个设计目标：

1. 核心目标：简易，即要求较低的学习和上手门槛。
2. 路径节点、路径节点的前一个控制节点和路径节点的后一个控制节点没有共线的限制。
3. 路径节点到它的前一个控制节点的距离不必和路径节点到它的后一个控制节点的距离相等。
4. 没有额外的快捷键用来实现特定功能。

3.1.3 通用交互方式

如表 3-1 所示，本文遵循了大多数编辑器、软件通用的交互方式，减少了用户的学习、记忆成本，使得其更易上手使用。

表 3-1 通用的交互方式

| 操作 | 交互方式 |
|-----------|---|
| 插入节点 | 当编辑器处于插入状态的时候，新插入的路径节点会跟随鼠标指针位置，如果有控制节点的话，控制节点也会相应的改变自己的位置。当单击鼠标左键的时候确定节点位置，插入结束。 |
| 移动节点 | 按下鼠标左键选中节点，按住不放，节点会跟随这鼠标指针的移动而移动。当松开鼠标左键的时候移动结束。 |
| 闭合路径 | 当插入节点时，鼠标指针移动到现有路径节点位置时，会出现闭合路径的提示，单击鼠标左键时不会插入这个节点，而是连接到鼠标指针所在位置的现有节点。 |
| 显示菜单 | 单击鼠标右键显示操作菜单。这个菜单会跟随这单击的对象而进行改变。 |
| 执行菜单栏中的操作 | 所有操作命令按照功能合理分组放置在菜单栏中，用户根据操作名称在菜单栏的下拉菜单中单击即可执行。 |

| | |
|-----------|--|
| 执行工具栏中的操作 | 操作命令按照功能合理分组放置在工具栏中，用户单击操作图标即可执行 |
| 通过快捷键执行操作 | `Ctrl+Z` 撤销，`Ctrl+X` 剪切，`Ctrl+C` 复制，`Ctrl+V` 粘贴，`Delete` 删除，`Ctrl+S` 保存。 |
| 悬停对象 | 当鼠标悬停在某个对象上时，这个对象会处于悬停状态，会给用户一个悬停反馈（如变为悬停色）。 |
| 选中对象 | 单击鼠标左键是会选中悬停的对象，同时显示与当前对象相关的对象（如果有的话）。 |

3.1.4 核心交互设计

路径编辑器的核心功能就是编辑路径，而路径则是由一系列节点组成，编辑路径实际上就分为插入新节点和移动现有节点两个核心操作。这两个操作不能出现在同一时刻，所以编辑交互设计的核心就是插入操作和移动操作如何进行相互切换的设计。而本文所设计的插入模式和移动模式切换的准则就是：插入一段路径，就立即调整好一段路径。以用户视角来看的操作逻辑是，我新插入一段路径后，大概率要对路径的形状进行调整，所以不再进行插入新节点。也就是说完成插入新节点后不会立即进行新节点的插入，而是需要手动操作进行新节点的插入。不过这个操作成本很低，无需快捷键等，只需要单击路径末端节点即可。

插入新节点。插入新节点有两种，一种是插入路径上的第一个节点，另外一种是在插入路径上的后续节点。插入节点类型的不同，交互方式也不同。

1. 插入路径的第一个节点。在空白处单击鼠标右键 → 选择“新建路径”选项，或者单击新建路径的工具栏按钮。
2. 插入路径上的后续节点。单击路径上的最后一个节点即可插入后续节点。在用户界面设计部分，会突出显示路径上的最后一个路径节点，来提醒用户这是路径的末端节点，单击即可插入后续节点。

插入新节点的交互方式会引入另一个难题：在左键单击路径末端节点的时候，有可能是插入新节点，也有可能是移动节点，那么该如何区分呢？解决方法一：末端节点只能用来插入新节点。解决方法二：根据按下鼠标左键后移动的距离，如果移动的距离在一定阈值内则判定为是插入新节点，否则就判定为是移动。

如果末端节点是无法移动的则显然破坏了整体上交互的一致性，而且特例会

给用户造成额外的记忆负担，所以本文采用了解决方法二。

插入新节点完成之后的状态转换。情景一：插入的是路径上的第一个节点。由于此时路径上只有一个节点，无法构成一条路径，也无法调整路径的形状，所以默认接着插入新节点。情景二：插入的不是路径上的第一个节点。由于新插入的节点和前一个节点构成了一条路径，用户可能需要调整控制节点来调整路径的形状，所以停止进行新节点的插入。

闭合路径。在插入新节点的时候，单击路径上的最后一个节点则会闭合路径。当将要闭合路径的时候，最后一个节点则会做突出处理，来提醒用户这个操作不是插入路径，而是闭合路径。

总而言之，本文设计的编辑交互只有单击路径最后一个节点来插入新节点需要记忆，其他的基本符合人们的直觉，降低了上手门槛和使用难度。其次没有插入新节点和移动节点之间的切换，去除了共线和等距的限制，少了很多的操作步骤，提升了使用效率。图 3-2 展示了路径编辑器核心功能的有限状态机。

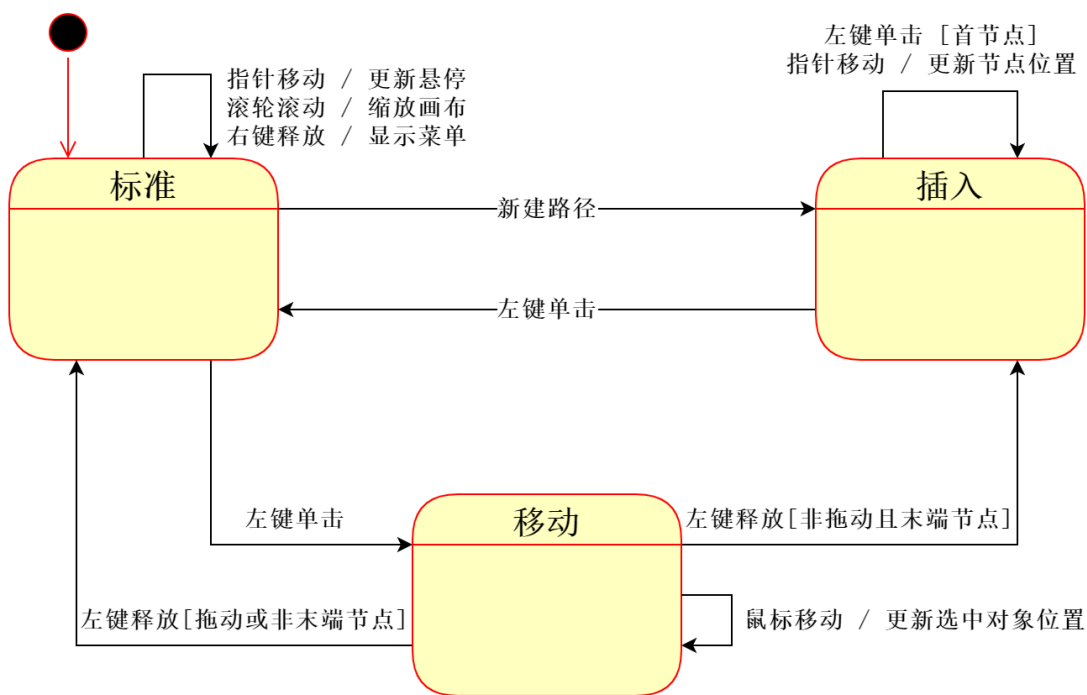


图 3-2 路径编辑器核心功能的状态机

3.1.5 其他功能交互设计

除了编辑路径这一个核心的功能，还有其他对用户体验很重要的功能。

画布的缩放和移动。画布的移动和缩放在编辑器中是一种很常见的功能，可以

将画布调整到合适的大小合适的位置来更好地编辑用户需要编辑的内容。当用户处理某个小细节的时候，需要缩放画布，并将画布平移到对应位置，然后进行调整。画布的缩放的交互采用的是鼠标滚轮的滚动，比较符合用户的使用习惯，容易记忆。移动画布采用右键拖动的方式来实现，可能左键拖动比右键拖动更符合用户的使用习惯，但是在经过在用户反馈负反馈当使用左键拖动画布的时候，因为节点比较小，移动节点容易误操作拖动画布，所以左键拖动画布的方案 没有被采用。使用快捷键+左键拖动画布的方式因为没有直接右键拖动画布的方式更方便，同时在记忆成本上也没有显著优势，所以也没有被采用。下面是本文所采用的交互方式：移动：使用鼠标左键拖动画布的空白处；缩放：鼠标滚轮滚动，依据滚动方向的不同来决定是放大还是缩小。

悬停和选中。当鼠标指针悬停在某个对象（节点、图片、路径等）上面时，软件应该给予用户适当反馈，这些反馈能够提示用户下一步可能进行的操作，放置用户的误操作。表 3-2 列举出了不同对象类型及其对应的悬停反馈。对于路径节点和控制节点来说，颜色会变为悬停色。对于路径来说，整条路径的颜色会变为悬停色。对于图片来说会出现悬停色的控制框（无控制顶点）。当鼠标悬停在某个非空对象上时，单击鼠标左键则表示选中，选中的对象会有相关反馈，最基本的就是颜色的变化，对于路径节点来说是选中的路径节点和其相邻的三次贝塞尔曲线都变为选中色。由于本文采用了先选中后操作的基本逻辑。所以只有在对象被选中之后，才会显示与其编辑相关的对象。比如选中路径节点时，会显示前一段路径上的 控制节点和后一段路径上的控制节点，同时被选中的路径节点和其相邻的三次贝塞尔曲线都变为了选中色。选中路径时，会显示路径的控制框，可以用来调整路径的大小。选中图片时，会显示图片的控制框，可以用来调整图片的大小。此时若鼠标左键进行单击则表示选中，也会有对应的反馈，对于路径节点来说，则是显示相关的控制节点，对于图片和路径来说则是显示控制框。

表 3-2 不同对象类型的悬停反馈

| 对象类型 | 悬停反馈 |
|-----------------|--|
| 空白 | 无 |
| 路径节点 | 路径节点颜色变为悬停色 |
| 控制节点 | 控制节点颜色变为悬停色 |
| 路径 | 整条路径颜色变为悬停色 |
| 图像 | 图像周围出现悬停色的控制框（无控制顶点） |
| 路径/图像的上/下控制框线 | 鼠标指针样式变为  |
| 路径/图像的左/右控制框线 | 鼠标指针样式变为  |
| 路径/图像的左上/右下控制顶点 | 鼠标指针样式变为  |
| 路径/图像的右上/左下控制顶点 | 鼠标指针样式变为  |

控制节点的显示。默认情况下是不显示控制节点的。只有在选中路径节点的时候，才会显示前一条路径上的和后一条路径上的控制节点。注意没有显示所有的控制节点，只显示了必要的，即用户可能会操作的控制节点，这是因为在节点较多是，显示所有控制节点以及对应的控制手柄占用了大量的视觉空间，且容易误操作，所以只显示了必要的控制节点以及对应的控制手柄，使得界面更简洁，更易用。

路径、图片的大小调整。选中路径/图片之后会显示相应的控制框，拖动控制框的四条边可以调整路径/图片的宽度/长度，而拖动控制框的四个顶点，则可以等比例的缩放路径/图片。

右键菜单的显示。右键单击时，会弹出相应的菜单，根据单击的对象不同，弹出的菜单也不同。表 3-3 给出了右键单击对象记忆弹出菜单的内容。

表 3-3 右键单击对象及其弹出菜单内容

| 右键单击对象 | 弹出菜单内容 |
|--------|--------------------------|
| 空白画布 | 新建路径，此处粘贴 |
| 图像 | 锁定、解锁、上移一层、下移一层、剪切、复制，删除 |
| 路径 | 剪切、复制，删除 |

预览模式。单击预览模式工具栏按钮，进入预览模式。

移动路径的时候不显示路径节点。因为移动路径的时候用户不需要对路径进行编辑，不显示路径节点可以减少用户的视觉负担。

3.2 路径编辑器的界面设计

3.2.1 布局

对于用户界面的设计应充分考虑用户对大多数应用程序所熟悉的操作习惯，这样用户可以运用已有的知识来使用界面^[13]。所以本系统采用了大多数应用程序所使用的布局方式，尽量不给用户造成新的认知上的负担，符合前面提到的简单易上手的目标。如图 3-3 所示，从上到下依次为标题栏，菜单栏，工具栏，路径编辑器的主工作区。

一个排列有序、错落有致的分组界面，能够让用户使用过程中得心应手^[13]。如图 3-3 所示本系统对菜单栏和工具栏中的功能进行分组，工具栏中不同分组之间用竖线分隔。通过分组，用户可以更高效地找到所需的功能。

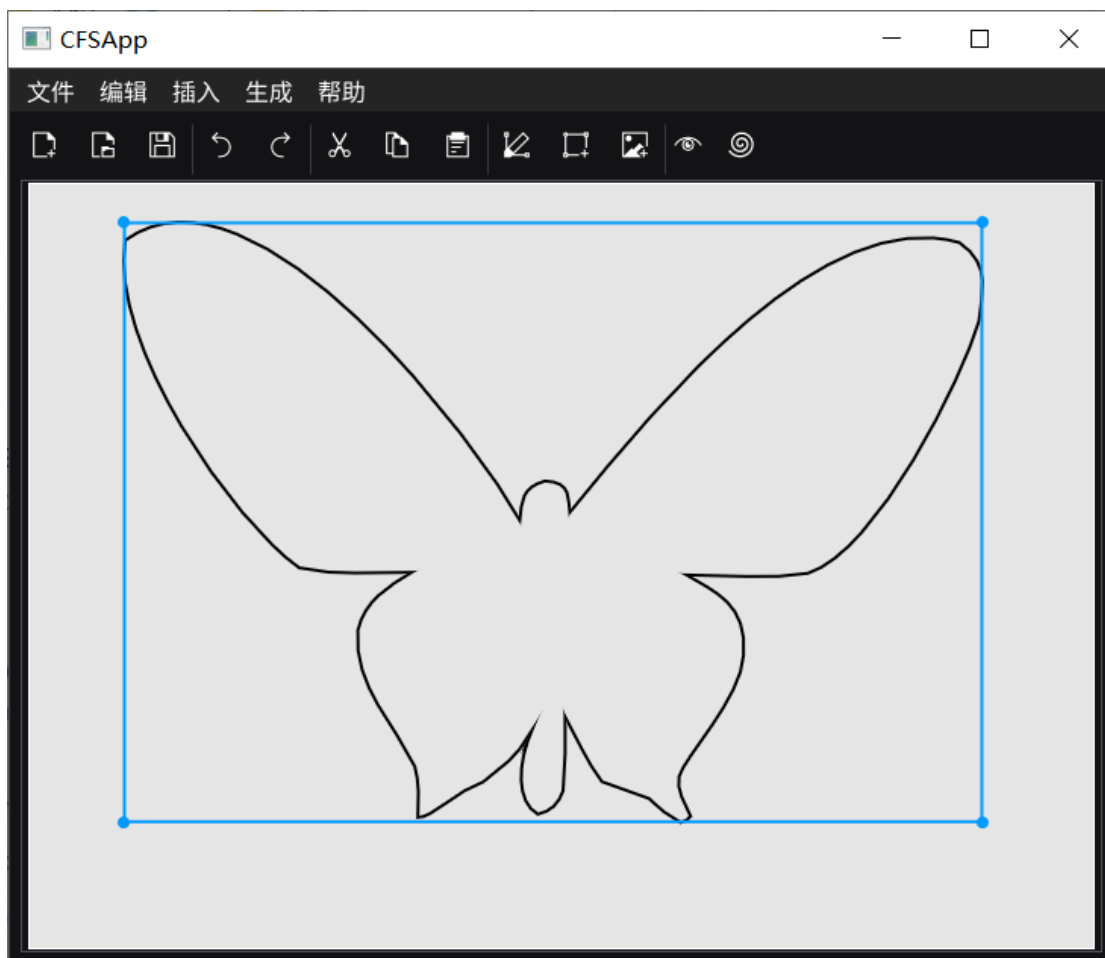


图 3-3 路径编辑器的用户交互界面










3.2.2 颜色

在人机交互界面色彩的选择上面，背景不宜使用醒目的色彩，需要突出强调的才需要使用鲜艳的色彩，若全部使用鲜艳的色彩则无法达到强调的效果^[14]。

所以，如图 3-3 所示，除了路径编辑器主工作区和可视化路径的颜色是浅色系外，其他的颜色都是深色系。这样使得主要与用户交互的内容被突出显示。

表 3-4 给出了路径编辑器不同部分对应的颜色设计。

表 3-4 路径编辑器的颜色设计

| 名称 | 颜色 | RGB 值 |
|--------------|---|---------------|
| 菜单栏的背景色 |  | 36, 36, 36 |
| 菜单栏中文字的颜色 |  | 255, 255, 255 |
| 工具栏的背景色 |  | 19, 19, 22 |
| 工具栏中图标的颜色 |  | 255, 255, 255 |
| 主工作区背景色 |  | 229, 229, 229 |
| 路径和路径节点的颜色 |  | 0, 0, 0 |
| 控制节点和控制手柄的颜色 |  | 128, 128, 128 |
| 悬停色 |  | 51, 204, 255 |
| 选中色 |  | 0, 153, 255 |

3.2.3 图标

连通货马螺旋线路径交互式生成系统的工具栏中的功能以图标的功能进行展现，工具栏中有新建文件、打开文件、保存文件、撤销、重做、剪切、复制、粘贴、新建路径、插入路径、插入图片、预览模式、生成连通货马螺旋线功能按钮。所选功能图标来自 KDE 的 Plasma Breeze 视觉风格的图标^[15]，它是一种扁平化设计风格的图标。扁平化风格的设计抛弃了渐变、阴影、高光、纹理等拟真视觉效果，从而降低用户的视觉信息干扰^[16]。所选图标均和对应功能有联系，容易联想和记忆。尽管图标尽量采用与功能相近的图标，但本系统还是设计了提示功能，如图 3-4 所示，当鼠标指针移动到图标上面时，会有对应的功能提示。



图 3-4 图标的提示文字

表 3-5 给出了本文采用的工具栏图标以及对应的功能。

表 3-5 工具栏图标以及对应功能

| 图标 | 功能 | 图标 | 功能 |
|---|------|---|---------|
|  | 新建文件 |  | 打开文件 |
|  | 保存文件 |  | 生成费马螺旋线 |
|  | 撤销 |  | 重做 |
|  | 剪切 |  | 复制 |
|  | 粘贴 |  | 新建路径 |
|  | 插入路径 |  | 插入图片 |
|  | 预览模式 | | |

3.3 路径编辑器的具体实现

3.3.1 悬停

判断鼠标指针是否悬停在某个对象上面的方法依据对象的不同而不同。表 3-6 给出了不同对象类型及其对应的悬停判定方法，除了图像和空白类型外，其余对象类型都是使用与鼠标指针之间的距离来判断是否悬停的。

表 3-6 对象类型及其悬停判定方法

| 对象类型 | 悬停判定方法 |
|--------------------|---|
| 空白 | 默认，即没有悬停任何对象 |
| 路径节点 | 鼠标指针到节点的距离小于点阈值 |
| 控制节点 | 鼠标指针到节点的距离小于点阈值 |
| 路径 | 鼠标指针到路径中的所有三次贝塞尔曲线的距离的最小值小于给定曲线阈值 |
| 图像 | 鼠标位于图像内部 |
| 路径/图像的上/下框线 | 鼠标指针的 x 坐标位于路径/图像的两个上/下顶点的 x 坐标之间，并且鼠标指针到框线的距离小于线阈值 |
| 路径/图形的左/右框线 | 鼠标指针的 y 坐标位于路径/图像的两个左/右顶点的 y 坐标之间，并且鼠标指针到框线的距离小于线阈值 |
| 路径的左上/右上/左下/右下控制顶点 | 鼠标到路径左上/右上/左下/右下控制顶点的距离小于点阈值 |
| 图像的左上/右上/左下/右下控制顶点 | 鼠标到图像左上/右上/左下/右下控制顶点的距离小于点阈值 |

根据对象形状的不同，用来悬停判定的距离阈值也不同。表 3-7 给出了悬停判定使用的所有距离阈值。因为节点显示的面积比较小容易误操作，所以给了比较大的阈值，而直线（主要是控制框线）和曲线（主要要是三次贝塞尔曲线）相对于点来说面积更大，更容易悬停，所以设置了相对较小的悬停阈值。

表 3-7 悬停判定的距离阈值

| 阈值 | 用途 | 默认值 |
|------|-----------------|------|
| 点阈值 | 判断鼠标是否悬停在点上的阈值 | 8.0f |
| 直线阈值 | 判断鼠标是否悬停在直线上的阈值 | 4.0f |
| 曲线阈值 | 判断鼠标是否悬停在曲线上的阈值 | 4.0f |

若一个鼠标指针的位置同时满足多种对象类型的悬停条件，则判断悬停的优先级如下：

1. 控制节点
2. 路径节点

3. 路径的控制框顶点
4. 路径的控制框线
5. 图像的控制框顶点
6. 图像的控制框线
7. 路径
8. 图像

对于根据距离是否小于特定阈值来判断是否悬停的对象，如果同种对象由多个满足阈值条件，则选与鼠标指针距离最近的一个作为悬停对象。

对于图像这种对象，如果有多个图像都满足悬停条件，则选最上层的图像作为悬停对象。

距离计算。对于大多数的对象而言，判断是否悬停主要是根据与鼠标指针之间的距离。根据对象的不同，计算大致分为以下几类：

1. 点到鼠标指针的距离
2. 水平或垂直的线段到鼠标指针的距离
3. 三次贝塞尔曲线段到鼠标的距离

点到鼠标的距离使用的是欧几里得距离。鼠标指针到水平线段的距离（前提是线段左端点 x 坐标 \leq 鼠标指针 x 坐标 \leq 线段右端点 x 坐标）直接用 $|\text{鼠标指针 } y \text{ 坐标} - \text{水平线段的 } y \text{ 坐标}|$ （坐标差的绝对值）计算获得。鼠标指针到垂直线段的距离同理。

计算鼠标指针到三次贝塞尔曲线的距离所使用的方法来自论文 "Improved Algebraic Algorithm On Point Projection For Bezier Curves" ^[17]：

1. 将计算距离问题转化成为了方程求根问题
2. 通过 Sturm sequence 方法^[18]求得方程的所实根的隔离区间。
3. 根据每个区间端点处中距离平方的导数进行分类，过滤掉不符合条件的区间，达到剪枝的目的。
4. 对于剩余的区间，结合二分法和牛顿法计算唯一的根
5. 选择距离最小的那个根

这种方法通过剪枝，去除掉了不需要计算的区间，减少了不必要的计算，同时也减小的根所在的区间范围，使得牛顿法更快更容易找到唯一解。这是一种快速且

高效的算法，在实现了这种算法之后，从鼠标移动，到出现悬停反馈，做到了用户无延迟感知。

3.3.2 选中

当鼠标左键单击时，当前悬停对象则会被选中，出现选中的一些与当前选中对象相关的反馈。有一个例外是，控制节点，控制框线，控制框线顶点是不能被选中的，即使单击了这些对象也不会改变选中对象，这是因为这些对象本来就是选中其他某个对象时出现的反馈对象，是用来调节反馈对象的，如果可以选中，则会更新选中对象，导致无法调节原选中对象。

3.3.3 平移和缩放

平移和缩放是一个坐标变换的过程，实际存储在数据结构中的坐标是位于原坐标系中的坐标，经过坐标变换后得到的新坐标系中的坐标，这个新坐标系中的坐标才是展示在屏幕上的坐标

设平移量为 \mathbf{t} ，缩放量为 s ，原坐标系中的坐标点为 \mathbf{p} ，对应的新坐标系中的坐标为 \mathbf{p}' ，则如公式 3-2 所示，原坐标系的坐标点到新坐标系中的坐标点的转换关系为：

$$\mathbf{p}' = s\mathbf{p} + \mathbf{t} \quad (3-2)$$

当鼠标指针拖动的时候只需要更新平移量 \mathbf{t} 就行了。

但是当鼠标滚轮滚动的时候，不能仅仅更新缩放量，因为要实现以鼠标指针为中心的缩放，而不是以原点为中心的缩放。具体实现就是相当于先将整个坐标系进行平移使得鼠标指针位于坐标原点处，然后执行缩放，最后再平移回去。设置缩放常量为 c ，鼠标指针位置为 \mathbf{p}_m ，缩放前的坐标点为 \mathbf{p} ，则如公式 3-3 所示：

$$((s\mathbf{p} + \mathbf{t}) - \mathbf{p}_m)c + \mathbf{p}_m = s\mathbf{c}\mathbf{p} + \mathbf{c}\mathbf{t} + (1 - c)\mathbf{p}_m \quad (3-3)$$

可得：

$$s' = sc \quad (3-4)$$

$$\mathbf{t}' = \mathbf{c}\mathbf{t} + (1 - c)\mathbf{p}_m \quad (3-5)$$

当放大时，缩放常量取 1.05，当缩小时，缩放常量取 0.95。

3.3.4 状态机

因为路径编辑器的状态转换比较复杂，如果只用 `if else` 语句实现会导致代码块缩进太深，实现逻辑不容易理解，因此本文使用了 `sml` 库，`sml` 库是 C++ 的一个高效的、扩展性强的状态机库^[19]。

3.3.5 导出、导入路径

在前面需求分析的章节中我们提到了路径编辑器导出的路径文件格式要尽可能与其他专业矢量图编辑软件所互通。经过调研，本文选择了 SVG 格式作为路径的导出文件格式。SVG 的全称是可扩展矢量图形（Scalable Vector Graphics），它是一种基于 XML 的用来描述二维矢量图形的标记语言^[20]。SVG 已经广泛应用于各种矢量图形编辑器，如 Adobe Illustrator, Inkscape 等。

导出路径。因为路径是以一系列三次贝塞尔曲线的节点的方式存储在内存中的，一个三次贝塞尔曲线的最后一个节点同时是其后一个三次贝塞尔曲线的一个节点。在导出为 SVG 时，主要用到了 SVG 的 `<path>` 标签，其内部是原生支持三次贝塞尔曲线的，所以在导出是只要按照 SVG 的语法保存文件即可。因为 SVG 是基于 XML 的，所以在保存为 SVG 时用到了 `tinysql2`^[21] 这个库在操纵 XML。

路径在内存中保存的是一系列三次贝塞尔曲线的节点，由于我们希望在软件退出后所编辑的路径能够不丢失，这时候就需要导出这个功能。

导入路径。因为本文采用 SVG 作为路径的存储格式，所以导入路径需要做的就是解析 SVG 文件，然后将其还原成为存储在内存中的数据结构。本文使用 NanoSVG^[22] 对 SVG 文件进行解析，然后将其还原成为存储在内存中的数据结构。

第 4 章 连通费马螺旋线的可视化

4.1 连通费马螺旋线可视化的交互设计

当用户在路径编辑器中完成编辑后，鼠标左键单击工具栏中的生成连通费马螺旋线路径按钮（工具栏最后一个图标）或者通过菜单栏中的生成→生成连通费马螺旋线后，如图 4-1 生成 CFS 对话框所示，会弹出生成连通费马螺旋线对话框，提示用户输入刀具路径的宽度。输入刀具路径宽度有两种交互方式，一种是粗略的：点击加/减按钮来改变刀具路径宽度，每次点击加/减按钮来增加/减少数值的为 1；一种是精确的：手动输入刀具路宽度。在输入刀具路径宽度后，点击生成按钮就会开始生成连通费马螺旋线路径。

等待生成完毕之后，如图 4-2 所示，会从主界面的右侧弹出可视化窗口，拖动两个窗口的分割线可以调整两个窗口的窗口大小。默认的可视化类型是动画，可以通过可视化类型下拉菜单选择可视化类型，有动画、仿真、三维三种。

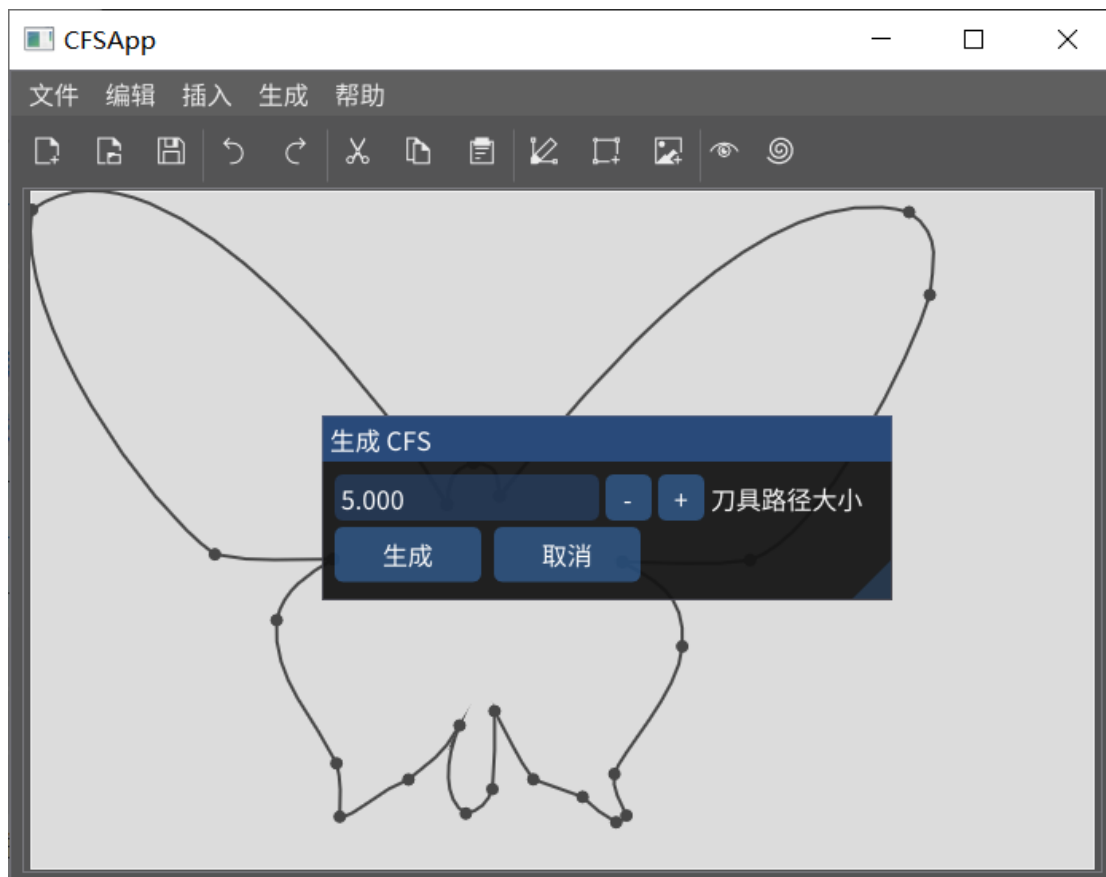


图 4-1 生成 CFS 对话框

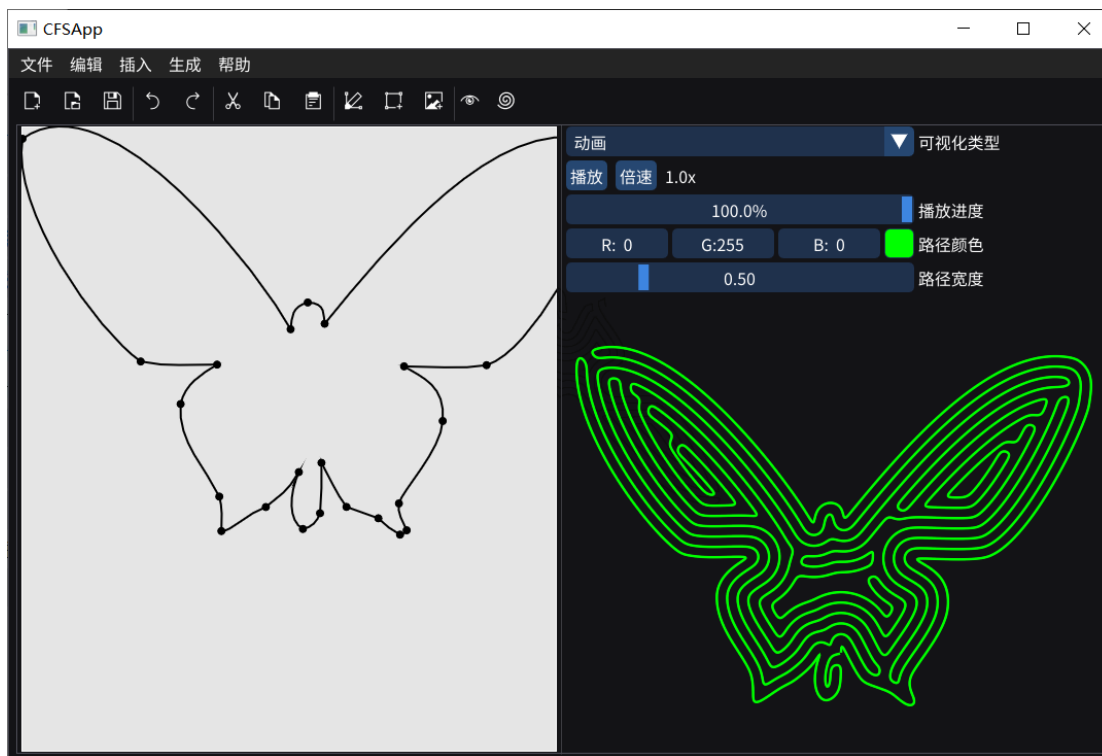


图 4-2 连通货马螺旋线可视化：动画

4.1.1 可视化类型：动画

播放。图 4-2 显示了动画可视化类型下的可视化界面，用户可以点击播放按钮播放整条连通货马螺旋线路径，路径会从起点开始一直延伸到路径的终点。

倍速。播放的时间与路径的长度无关，默认的播放时常都是 15 秒。如图 4-3 所示，当鼠标单击倍速按钮时，会弹出播放速度的候选框，有 0.5x, 1.0x, 1.5x, 2.0x 四个倍速选项，用户可以通过这些选项设置播放的速度，同时倍速按钮右侧会显示当前播放的倍速。

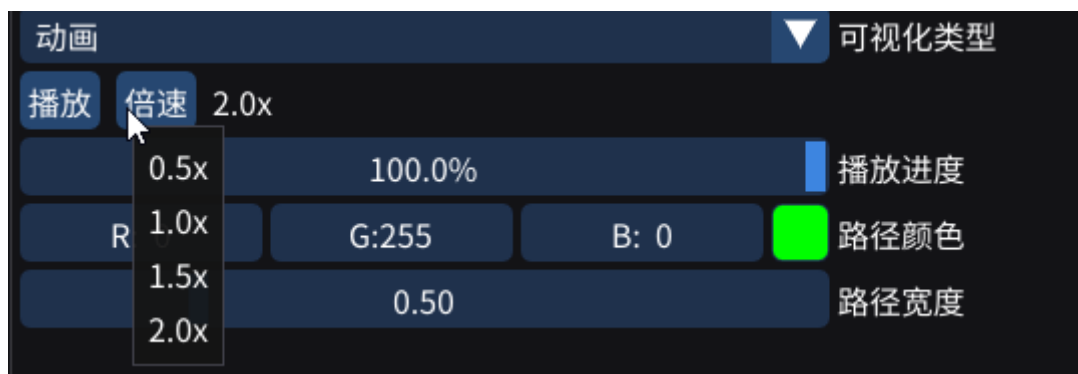


图 4-3 调整播放倍速

播放进度。用户可以通过播放进度条查看当前的播放进度。除此之外，用户还

可以手动拖动播放进度条停留在某个播放进度。图 4-4 显示了当播放进度为 50% 时的连通费马螺旋线路径。



图 4-4 播放进度为 50% 时的连通费马螺旋线路径

路径颜色。用户可以拖动 R, G, B 三个滑块，来调整路径颜色的 R, G, B 的值。如图 4-5，用户也可以单击最右边的颜色方块，弹出色板，进行颜色选择。

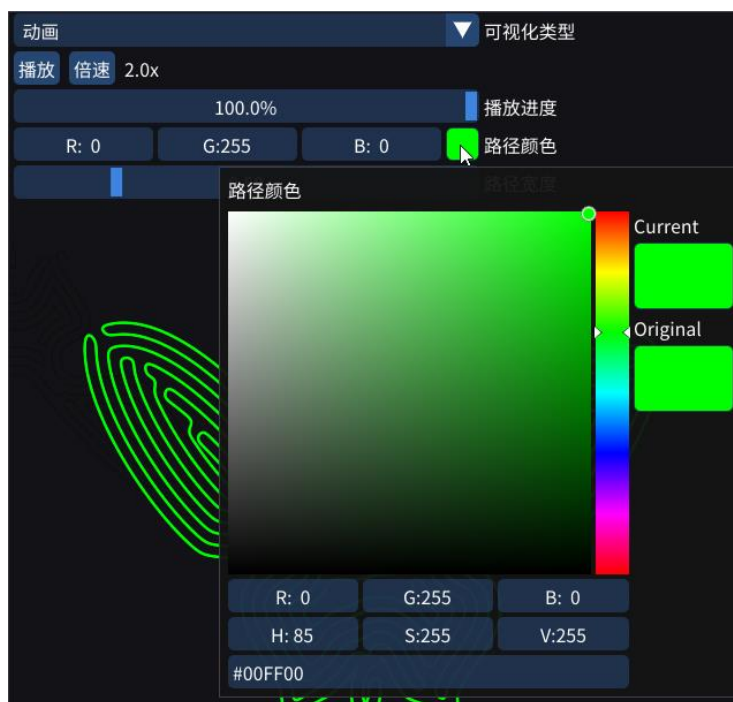


图 4-5 调整路径颜色

路径宽度。用户可以拖动路径宽度的滑块，来调整路径的宽度。

平移和缩放。用户可以通过鼠标指针拖动来平移移动整个画布，通过鼠标滚轮滚动来缩放整个画布。

4.1.2 可视化类型：仿真

在仿真模式下，用户可以清晰地看出欠填充和过填充的区域。

路径颜色。调整路径颜色的交互方式与动画模式下的交互方式是相同的，因此不再进行赘述。

移动和缩放。与路径编辑器中的移动和缩放交互方式相同，不再进行赘述。

4.1.3 可视化类型：三维

使用 Arcball 的交互方式来旋转三维模型。Arcball 是一种用于三维计算图形的输入技术，使用鼠标来调整对象的空间方向^[23]。

4.2 连通费马螺旋线可视化的界面设计

4.2.1 布局




动画和仿真可视化界面与路径编辑器的编辑区域是左右分栏的，可视化界面会在生成连通费马螺旋线后在路径编辑区域的右侧弹出，左右分栏的大小是可以调整的。左右分栏的好处是可以同时看到生成连通费马螺旋线算法的输入和输出，便于用户根据输出调整输入。而三维可视化由于技术原因只能占据整个窗口，通过你是提供一个浮动的窗口用来切换可视化类型

4.2.2 颜色

连通费马螺旋线可视化的颜色同样遵循背景色采用深色，而连通费马螺旋线采用亮色，用来突出可视化的连通费马螺旋线路径。

表 4-1 给出了可视化界面中不同部分对应的颜色。

表 4-1 可视化的颜色设计

| 名称 | 颜色 | RGB 值 |
|-----------|---|-------------|
| 背景色 |  | 19, 19, 22 |
| 动画模式下路径颜色 |  | 0, 255, 0 |
| 仿真模式下路径颜色 |  | 0, 255, 255 |

4.3 连通费马螺旋线可视化的具体实现

4.3.1 拉普拉斯平滑

生成的连通费马螺旋线路径的平滑性是一个很重要的因素，会直接影响到打印的质量，为了提升生成的连通费马螺旋线路径的平滑性，本文采用拉普拉斯平滑算法对生成的结果进行平滑：

设生成的连通费马螺旋线路径是由 n 个离散的点 $\mathbf{p}_0 \dots \mathbf{p}_{n-1}$ 表示的路径，则在经过一次拉普拉斯平滑后：

$$\mathbf{p}'_0 = \mathbf{p}_0 \quad (4-1)$$

$$\mathbf{p}'_{n-1} = \mathbf{p}_{n-1} \quad (4-2)$$

$$\mathbf{p}'_i = \frac{1}{2}(\mathbf{p}_{i-1} + \mathbf{p}_{i+1}), i = 1 \dots n-2 \quad (4-3)$$

会得到一个新的由 n 个离散的点 $\mathbf{p}'_0 \dots \mathbf{p}'_{n-1}$ 表示的路径，这个路径比原路径要更平滑一些，因为算法生成的连通费马螺旋线路径的点距较小，点比较密集，所以可以进行多次拉普拉斯平滑；在通过多次实验之后，考虑平滑效果和平滑耗时，最终采取了对生成后的连通费马螺旋线进行 20 次拉普拉斯平滑的方案。

4.3.2 路径采样

路径采样后获得路径上的一系列离散的点，用来作为连通费马螺旋线路径算法的输入。

这里的路径指的是作为连通费马螺旋线路径算法输入的路径，位于路径编辑器中，以多段连接的三次贝塞尔曲线构成的拓扑联通区域。

将路径转换为离散的点，实际上是将多段三次贝塞尔曲线转换为离散的点。如公式 4-4 所示，设三次贝塞尔曲线路径的表达式为：

$$\mathbf{p}(t) = (1-t)^3 \mathbf{p}_0 + 3(1-t)^2 t \mathbf{p}_1 + 3(1-t)t^2 \mathbf{p}_2 + t^3 \mathbf{p}_3, t \in [0, 1] \quad (4-4)$$

其中 p_0, p_1, p_2, p_3 为三次贝塞尔曲线路径的四个节点。

本文实现了等距和非等距两种将路径转为离散的点的方法。

非等距采样的实现方法。首先计算出三次贝塞尔曲线路径的长度 l ，然后根据刀具路径大小 s 计算出这段三次贝塞尔曲线路径上需要采样的点数 $c = \frac{l}{s}$ ，最后就能计算出相邻采样点之间的间隔实现 $t_{interval} = \lceil \frac{1}{c} \rceil$ ，这样第 i 个采样点的时间 $t_i = (i - 1)t_{interval}$ ，带入公式 4-4 中，即可得出第 i 个采样点的坐标。非等距的方法在曲线曲率比较高的地方点比较密集，而在比较平滑的地方点比较稀疏，这样的好处是能够最大地保证不丢失曲线路径的形状。因为点距是不固定的，有可能会超过刀具路径的宽度，所以要在距离超过刀具路径宽度的两个点之间进行插值。

等距采样的实现方法，参考“Moving Along a Curve with Specified Speed”^[24] 这篇文章的方法。等距采样对公式 4-4 进行了换参数操作。即求路径长度为 d 的点对应的 t ，将其转换成了偏微分方程求根问题，采用了龙格库塔方法进行求解即可以得到路径长度为 d 时对应的 t 值。

4.3.3 平移和缩放

平移和缩放的实现与路径编辑器实现的方法相同，就不再赘述了。

4.3.4 路径动画化

路径动画化是为了模拟打印喷嘴的真实移动过程，可以直观的展示一条连通费马螺旋线路径是如何被打印出来的。

因为本文采用了 ImGui 来实现 UI 和交互，所以在实现动画化的方式与一般的 RMGUI 不同。

当点击播放按钮时，设置播放进度为 0%，刷新时间为当前时间，设置当前状态为动画中。用播放总时间为 15s 计算出单位播放进度所需要的时间 δ ，在绘制每一帧的时候，判断刷新时间是否小于当前时间，如果小于当前时间则加一个单位播放进度，然后刷新时间加上 δ ，重复此步骤，直到刷新时间大于等于当前时间或者播放完成。当播放完成时，设置当前状态为非动画中。

4.3.5 欠填充和过填充可视化

因为如果要真实模拟现实中的欠填充和过填充，需要较深的物理知识，超出了本文的范围。所以本文采用了图形上的方法来模拟现实中的欠填充和过填充。

首先假设打印喷嘴所打印出来的刀具路径是等宽的，欠填充区域是指指定宽度的刀具路径没有覆盖到这个区域，而过填充的区域刀具路径会多次经过，经过区域的次数越多则表明这个区域的过填充的程度越高，本文假设过填充的程度与刀具路径经过的次数成正比。

基于以上假设，本文提出了一种模拟欠填充和过填充可视化的一种简单的方法：使用有宽度的线条表示刀具路径，同时线条的透明度值设为 0.33，用线条透明度值的叠加去模拟过填充，比如如果线条经过一个区域 2 次，则透明度值就为 $0.33 \times 2 = 0.66$ ，3 次就是 0.99。通过实践表明几乎很少有区域被经过 3 次以上，所以基准透明度设置为 0.33 是比较合理的。这样就可以通过颜色的透明度就可以分辨出过拟合的区域，颜色越不透明则表示这片区域就越过拟合。如果一片区域没有被经过，则透明度值为 0，会直接显示背景的颜色，也非常容易分辨出来。

图 4-6 显示了一个连通货马螺旋线路径欠填充和过填充的可视化。

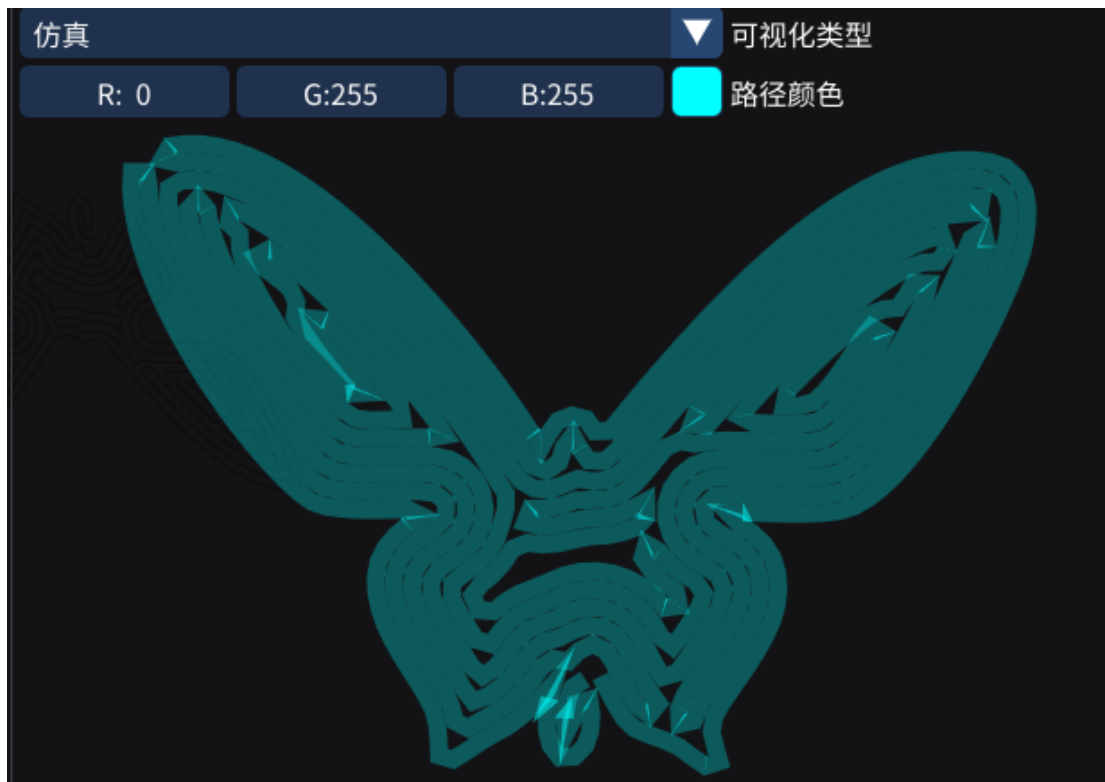


图 4-6 可视化类型：仿真

4.3.6 三维可视化

因为本文所采用的 ImGui 框架不提供加载三维模型的能力，所以本文使用 libigl^[25] 加载三维模型。因为 libigl 是独立与 ImGui 的，无法将渲染的三维模型渲染到 “ImGui 的窗口” 中，但好在 libigl 和 ImGui 都可以使用 OpenGL 作为渲染后端，可以复用 glfw 创建的窗口，所以本文采用当点击三维可视化模式的时候，隐藏掉路径编辑器和可视化界面，显示 libigl 渲染的三维模型，然后提供一个 “ImGui” 的窗口用于返回到路径编辑器和可视化界面。受限于 ImGui 框架，这种交互模式是与前两种可视化类型不同的。

在三维可视化类型下，用户可以拖动三维模型，以不同的视角去查看路径。图 4-7 显示了在三维可视化模式下的连通货马螺旋线路径。

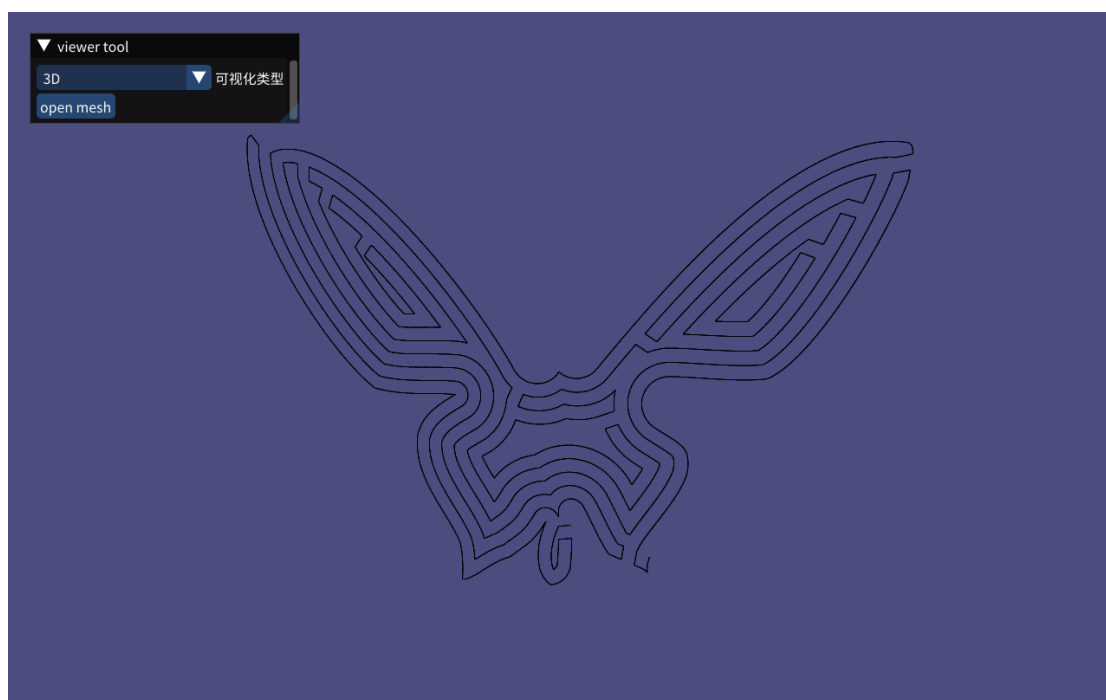


图 4-7 可视化类型：三维

第5章 结束语

5.1 本文总结

本文基于已有的连通货马螺旋线路径算法实现了连通货马螺旋线路径的交互式生成系统，其中实现了一个简单易用、功能全面的路径编辑器，用多段三次贝塞尔曲线表示路径，且根据连通货马螺旋线路径算法的特点优化了交互方式，除此之外还引入了根据图像编辑路径的交互方式，为用户提供了多种选择；此外，还实现了可视化系统，提供了动画、仿真、三维三种可视化模式，使得生成的连通货马螺旋线路径清晰明了，满足了用户的多种需求。总而言之，连通货马螺旋线路径系统大大降低了用户在使用连通货马螺旋线路径的使用门槛，提升了用户的使用效率。

5.2 工作展望

虽然本文完成的连通货马螺旋线路径的交互式生成系统的功能已经很完善易用了，但是仍有改进空间。

5.2.1 探索自动从图片中获取路径

本文所提供的依照图片生成路径的交互方式是手动的，随着人工智能技术的发展，在未来的工作中可以集成人工智能方法，自动检测图片中的路径，然后转换为路径编辑器所用的数据结构，这样将大大减少用户的手动工作量。

5.2.2 探索真实的物理模型

在仿真可视化中，是采用图像处理的方法来模拟连通货马螺旋线的欠填充和过填充，还有很大的改进空间。为了使仿真的效果更好，在未来的工作中，可以探索其真实的物理模型而不是简单地使用图像处理的方法来模拟连通货马螺旋线的欠填充和过填充，这样用户即使没有将连通货马螺旋线路径给打印出来也能够大概地了解连通货马螺旋线路径打印出来的质量好坏。

参考文献

- [1] Zhao H, Gu F, Huang Q X, et al. Connected Fermat Spirals for Layered Fabrication[J]. ACM Transactions on Graphics, 2016, 35(4):100.1-100.10.
- [2] 赵海森. 面向增减材料制造的几何研究与应用[D].山东大学,2018.
- [3] 李艳民. 基于 Qt 跨平台的人机交互界面的研究和应用[D].重庆大学,2007.
- [4] 李森林, 邓小武. 一种跨平台类库 WxWidgets 应用开发研究 [J]. 电脑与信,2009,No.157(03):32-34.
- [5] 李悦. 基于 Python+Tkinter 的 Linux GUI 辅助管理工具的设计与实现[D].吉林大学,2009.
- [6] Brendel, Felix, and Sven Liedtke. Exploring the immediate mode GUI concept for graphical user interfaces in mixed reality applications. GI VR/AR Workshop 2022. Gesellschaft für Informatik eV, 2022.
- [7] Sean Barrett. Immediate mode guis. Game Developer Magazine, 2005, 12:34–36.
- [8] Omar Cornut. About the ImGui paradigm[EB/OL]. (2023/4/29)[2023/4/26]. <https://github.com/ocornut/imgui/wiki/About-the-ImGui-paradigm>.
- [9] Ferraiolo J, Fujisawa J, Jackson D. Scalable Vector Graphics (SVG) 1.1 Specification. 2003.
- [10] 王继成, 高珍. 软件需求分析的研究[J]. 计算机工程与设计, 2002, 23(8):4.
- [11] 黄怡强,郭钦祥,黄怡胜.浅谈软件开发需求分析阶段的主要任务[J].中山大学学报论丛,2002(01):262-265.
- [12] Marschner S, Shirley P. Fundamentals of computer graphics. 5th edition.[M]. CRC Press, 2022.
- [13] 白文涛,刘正捷.用户界面的需求分析与设计原则[J].大连海事大学学报,2004(04):86-88.DOI:10.16411/j.cnki.issn1006-7736.2004.04.023.
- [14] 姜葳. 用户界面设计研究[D].浙江大学,2006.
- [15] Uri Herrera. Breeze Icons[EB/OL]. <https://invent.kde.org/frameworks/breeze-icons>
- [16] 马娜娜.简洁而不乏味——浅谈扁平化界面设计[J].大众文艺,2013,(16):137-138.
- [17] Chen X D, Yin Z, Shu Z, et al. Improved Algebraic Algorithm On Point Projection For Bézier Curves[C]// Proceeding of the Second International Multi-Symposium of Computer and Computational Sciences (IMSCCS 2007), August 13-15, 2007, The University of Iowa, Iowa City, Iowa, USA. IEEE, 2007.
- [18] Acton F.S., Witkins W. Numerical methods that work(2nd printing). Washington : Mathematical Association of America, 1990.
- [19] Kris Jusiak. SML: C++14 State Machine Library[EB/OL]. <https://github.com/boost-ext/sml>
- [20] 王仲,董欣,陈晓鸥.SVG——一种支持可缩放矢量图形的 Web 浏览语言规范[J].中国图象图形学报,2000,(12):71-75.
- [21] Lee Thomason. TinyXML2 is a simple, small, efficient, C++ XML parser that can be easily

integrated into other programs[EB/OL]. <https://github.com/leethomason/tinyxml2>

[22] Mikko Mononen. Simple stupid SVG parser[EB/OL]. <https://github.com/memononen/nanosvg>

[23] Shoemake, Ken. "ARCBALL: A user interface for specifying three-dimensional orientation using a mouse." Graphics interface. Vol. 92. 1992.

[24] Eberly D. Moving along a curve with specified speed. Geometric Tools, Redmond WA, USA, Tech. Rep. 2007.

[25] Alec Jacobson. Simple C++ geometry processing library[EB/OL]. <https://github.com/libigl/libigl>

致 谢

在本文完成之际，我要首先感谢赵海森教授，在本文写作和软件系统开发的过程中，赵海森教授给予了我很多宝贵的建议和指导。我还要感谢史子涵和乔婧两位同学对本文初稿提出了宝贵的建议。同时我还要感谢山东大学给我提供了一个良好的学习环境。最后我还要感谢我的父母一直无私地给我提供帮助。

光阴似箭，日月如梭。本文的完成，也预示着我的本科生活即将结束。回首本科四年的求学之路，虽大部分时间都在经历着疫情，但是授课的老师教授依然认真地通过网络授课，丝毫不受疫情的影响。正是由于老师们的认真负责，我才能在本科阶段的学习中能够学到很多知识，为我将来的道路打好了基础。在此我真诚地向所有给予过我帮助的老师、同学说声感谢！当然，我还要感谢我深爱的父母在我求学期间一直无私地给予我帮助和支持。

一个阶段的结束，也预示着另一个阶段的开始。愿自己在下一个阶段中能够永远保持初心，一路坚持走下去。

附录 1 译文中文

用于分层制造的连通费马螺旋线

摘 要

我们开发了一种新型的“空间填充”曲线，连通费马螺旋线，并将其引人注目的特性展示为分层制造的工具路径填充模式。与 Peano 或 Hilbert 曲线等经典空间填充曲线不同，它们不断缠绕和束缚以保持局部性，连接的费马螺旋线主要由长而低曲率的路径形成。这种几何特性连同连续性会影响分层制造的质量和效率。给定一个相连的二维区域，我们首先将其分解为一组子区域，每个子区域都可以填充一个连续的费马螺旋线。我们表明，始终可以在填充区域外边界上的大致相同位置开始和结束费马螺旋线填充。这个特殊属性允许沿着分解子区域的图遍历系统地连通费马螺旋线填充。结果是一条全局连续曲线。我们证明，与传统的填充图案相比，按照连接的费马螺旋线的工具路径打印二维层可以实现高效和高质量的制造。

1 简介

增材制造技术的出现 [Gibson et al. 2015] 导致计算机图形学界对 3D 制造的几何优化越来越感兴趣。最近许多尝试的重点是形状优化：如何最好地配置 3D 形状，例如，通过挖空或加固，以实现质量和成本效益的制造。在这项工作中，我们从不同的角度来看这个问题。我们没有解决打印什么的高级问题，而是检查与如何打印给定对象相关的较低级别但基本的问题。在最基本的层面上，增材制造或分层制造通过移动打印头来进行，打印头逐层挤出或熔合打印材料。打印每一层时，打印头按照规定的工具路径填充由打印对象的形状定义的二维区域。在拓扑上，工具路径的连续性对于制造至关重要。工具路径的不连续性或轮廓的多重性会迫使打印喷嘴进行开关切换，从而对构建质量和精度产生负面影响 [Dwivedi 和 Kovacevic 2004; 丁等。2014]。在几何上，急转弯和拐角是不受欢迎的，因为它们会导致层边界处的离散化伪影并导致打印头减速，从而降低打印速度和降低填充质量 [Jin 等人。2014]。

由于其简单性，之字形已成为当今 3D 打印机最广泛采用的填充图案

[Gibson 等人。 2015];有关各种填充图案,请参见图 2。然而,锯齿形填充由许多急转弯组成,当打印具有复杂边界或空心结构的形状时,这个问题会被放大。由欧几里德距离变换的等值线形成的等值线平行刀具路径提供了一种补救措施,但由于等值线彼此断开,因此会导致高轮廓复数。对于正方形等简单形状,螺旋填充图案是连续的。然而,对于更复杂的形状,轮廓平行填充和螺旋填充都倾向于留下与距离变换的奇点相对应的孤立“口袋”,如图 3(a) 所示。这些口袋是不连接的,导致路径多。一个有趣的几何问题是连接的二维区域是否总是可以由一个或多个螺旋形成的连续图案填充。

在本文中,我们介绍了使用费马螺旋线 [Wikipedia 2015] 作为基本的二维填充模式,并开发了一种基于连接的费马螺旋线或 CFS 的刀具路径规划算法,以连续填充连接的二维区域。费马螺旋是一种有趣的空间填充模式,具有两个交错的子螺旋,一个向内,一个向外;参见图 2(e)。费马螺旋线以前没有被用于刀具路径规划,它们具有三个关键特性,使它们作为填充图案具有吸引力:

与等高线平行路径一样,费马螺旋线符合区域边界,在中心有一个急转弯。

多个覆盖二维区域的费马螺旋可以连续连接。传统的螺线要么向内要么向外移动,而费马螺旋线先进后出,允许其中的几个在边界处连接。

费马螺旋线的起点和出口点可以在其边界上任意选择;参见图 2(ef)。这种特殊性质促进了费马螺旋线之间的连接。

我们开发了一种算法来构建 CFS 曲线以填充单连接的二维区域。首先,该算法将输入区域分解为一组子区域,每个子区域允许单个费马螺旋线连续填充;我们称这些子区域为可螺旋的。每个费马螺旋线的起点和出口点沿着螺线边界彼此相邻放置。然后,我们获得可螺旋子区域的连续遍历,并通过起点/出口点连接它们各自的费马螺旋,形成全局连续曲线。可以应用进一步的优化来提高公平性和间距。生成的曲线比之字形填充有更少的急转弯,并且主要由长而低曲率的路径组成。

我们展示了为具有不同外部和内部复杂性的输入形状构建的 CFS 曲线。我们使用 CFS 模式制造 2D 层和 3D 打印,并将结果在视觉和数字上与通过锯齿形和等高线平行工具路径进行的传统填充提供的结果进行比较。新的填充图案似乎擅长处理具有大量凹陷和许多内部孔洞的形状。

回想起来,我们从相连的费马螺旋线中寻求的理想性质几乎与希尔伯特曲线等经典空间填充曲线所具有的性质完全相反;参见图 2(c)。希尔伯特或皮亚诺曲线

被设计成缠绕和弯曲以保持空间遍历的局部性。CFS 曲线旨在尽可能避免弯曲以获得更高层次的公平性。通常，即使在无限分辨率下，CFS 曲线也不能保证完全覆盖任意 2D 区域。同样，我们的新曲线不具备经典空间填充曲线的递归特性。然而，对于分层制造，新曲线显然比分形填充图案更具吸引力。

2 背景及其相关工作

最近，计算机图形学领域的工作蓬勃发展，旨在优化 3D 形状或其配置以实现高效和有效的制造，例如，确保或提高物理稳定性 [Pre'vost 等人。 2013]，结构强度 [Stava 等人。 2012;希尔德布兰德等人。 2013;卢等人。 2014]，或外观 [Zhang et al。 2015] 的打印，以节省材料 [Vanek 等人。 2014b;胡等。 2014]，并适应有限的印刷量[Luo et al。 2012;瓦内克等人。 2014a;陈等。 2015;姚等。 2015]

在本节中，我们将重点关注增材制造 (AM) 背景下的刀具路径规划问题。首先，为了解释刀具路径的连续性和公平性的重要性，我们提供了一些关于控制沿路径挤出的粘弹性材料的喷嘴机构的背景材料，以及控制刀具路径的电机的机械结构。然后我们讨论优化刀具路径的相关几何和工程方法。我们的报道并不意味着详尽无遗。有兴趣的读者应该参考 Gibson 等人的书。 [2015]，[Kulkarni 等人的简短调查。 2000; 丁等。 2014]，以及 Dinh 等人最近的 SIGGRAPH 课程。 [2015]。

刀具路径连续性。熔融沉积成型或 FDM 是应用最广泛的增材制造技术。在 FDM 过程中，细丝熔化成粘弹性材料并从打印喷嘴的小开口中挤出。由于液体的可压缩性，通常很难预测粘弹性材料的排放量以创建连续的挤出控制。因此，来自喷嘴的灯丝的第一部分通常填充不足或填充过多。这种不均匀的填充在打印对象的表面附近出现时会导致视觉伪影。当它们出现在填充路径之间时，细丝之间的附着力会减弱，从而降低打印强度。关闭挤出时，进料电机停止与长丝挤出停止之间的时间间隔难以控制，这再次导致填充不均匀。当打印材料被打印头熔化时，也会出现类似的情况，例如，对于基于粉末的打印。同样，沿工具路径的任何不连续性都需要喷嘴移动，这对打印没有贡献。因此，设计刀具路径的主要目标是最大限度地减少沿路径的开/关切换，或者换句话说，最大限度地提高其连续性。

刀具路径几何。刀具路径的几何形状，尤其是它们的曲率，会影响制造时间和质量。与软转弯的情况相比，当刀具路径围绕急转弯自行转弯时，需要更多的减速

和加速时间，从而导致挤压头的减速更多。同样，锐角也会导致灯丝的过度填充或填充不足 [Jin 等人。 2014]。因此，没有急转弯的长而连续的工具路径可以使挤压头能够以接近允许的最高速度且变化很小的方式沿着整个工具路径移动，从而导致高效和高质量的制造。

方向平行与轮廓平行填充。商业 AM 系统中最流行的填充方法遵循之字形模式 [Ding 等人。 2014]。与光栅扫描一样，之字形属于方向平行填充的类别。相反，轮廓平行路径由一组平行于给定切片轮廓的闭合轮廓组成 [Yang et al。 2002]。在简单的 2D 区域上，与之字形相比，这样的路径会导致更平滑的转弯和物体边界 [El-Midany 等人。 1993]，但他们总是有很高的轮廓复数。还提出了混合填充 [Jin 等人。 2013]；在用之字形填充剩余的内部区域之前，它们会向内生成一些轮廓，但是两种填充图案之间的连接可能会变得可疑。当要填充的 2D 切片具有具有许多凹面的复杂形状时，两种填充模式的标准实现都容易出现不连续性问题。

螺旋刀具路径。螺旋刀具路径已广泛应用于型腔加工 [Ren et al. 2009]。 Held 和 Spielberg [2014] 将 2D 层分解为可螺旋形的口袋，并按照单独的经典螺旋图案加工每个口袋；没有构建全局连续路径。螺旋形工具路径对于 AM 来说不太常见，一个主要原因（也适用于轮廓平行填充）是由于缺乏方向偏差，相邻切片的螺旋图案相互复制并且不能以一定角度“交叉编织”；这可能会损害 FDM 打印机的制造强度 [Gibson 等人。 2015]。这个问题可以通过混合填充来解决，例如，在螺旋层和之字形层之间交替。我们的工作重点是优化螺旋填充的连续性。

空间填充曲线。填充二维区域的连续刀具路径是空间填充曲线 (SFC)。 SFC 已被用于各种应用，例如图像编码 [Dafner 等人。 2000] 和迷宫设计 [Pedersen and Singh 2006]。类似分形的 SFC 已被建议作为 AM 的填充模式 [Wasser 等人。 1999]。然而，它们实现起来很复杂，而且充满急转弯。工具路径填充问题与割草有一些相似之处 [Arkina et al. 2000]，它是在一个相当不同的设置下制定的：它为具有规定形状的“刀具”寻找最短路径以覆盖所有点（可能多次）在二维区域中。

迷宫。迷宫和许多著名的迷宫图案也充满了空间 [Wikipedia 2016]。单线迷宫曲线是连续的，起点和终点都在同一点，就像我们相连的费马螺旋线一样。 Pedersen 和 Singh [2006] 开发了一种随机曲线演化算法来产生富有表现力的空间填充迷宫，其中曲线粒子的布朗运动受到吸引力 - 排斥力以及局部公平性和场对齐约束。相比之下，我们的算法是自上而下的，并且在构造中具有更多的全局结构

控制。生成的曲线具有更平滑的边界和更少的转弯。

域分解。获得连续刀具路径的一种有趣方法是将 2D 区域分解为多个子区域，每个子区域允许连续填充。然后将这些区域填充连接起来以实现全局连续性。按照这些思路，[Dwivedi 和 Kovacevic 2004] 将多边形分解为单调的子多边形，并使用闭合的之字形曲线填充每个子多边形，沿着该曲线可以任意选择起点/入口点。丁等。[2014] 执行凸分解，对于每个凸多边形，找到最佳之字形方向以促进多边形填充之间的连续连接。然而，这两种方法都是为处理多边形输入而设计的，无法正确处理具有平滑凹形边界的形状。

我们的工作还依赖于区域分解，同时它可以处理任意二维形状作为输入。我们没有使用之字形，而是使用 Fermat 螺旋填充来实现连续性和更程度的公平性。分解方案旨在适应轮廓平行和螺旋刀具路径。

可连续填充的形状。在域分解方法中选择多边形凸性 [Ding et al.2014] 和单调性 [Dwivedi 和 Kovacevic 2004] 以实现刀具路径连续性，因为这两种形状属性都确保了锯齿形图案的连续填充。对于单调多边形，一组有限的扫描方向保证了这一点，而对于凸多边形，任何扫描方向都会导致连续的锯齿形填充。螺旋性是螺旋或等高线平行填充的对应物，据我们所知，这种形状属性以前没有被研究过。

3 螺旋、费马螺旋和螺旋性

在介绍螺旋线之前，我们首先研究等高线平行刀具路径并将它们与二维区域的欧氏距离变换相关联。然后，我们描述了如何将平行轮廓转换为规则间隔的螺旋图案并定义可螺旋性。最后，将螺旋填充转换为费马螺旋填充，我们可以在边界上任意选择起点和终点。轮廓平行路径作为等值线。让 R 成为一个连接的二维区域，其边界由 ∂R 表示。的欧几里德距离变换 ∂R 定义了 R 上的标量距离场 \mathcal{D}_R ，其中对于每个点 $p \in R$ ，是从 $\mathcal{D}(p)$ 到 ∂R 最短距离 p 。与距离相关联的等值线由 d 标量值为 d 的所有点组成 R ；边界 ∂R 是与等值 0 关联的等值线。根据制造过程中填充材料的宽度，这组等值线平行工具路径对应于一组等距的等值线，这些等值线都是断开的相互之间，如图 4(a) 所示。

螺旋性和螺旋性。两个相邻的等值线可以通过断开和重新路由等值线来连接以形成单个连续路径，如图 4(b) 所示。以这种偏移方式重新布置相邻轮廓将导致

螺旋图案。如果其中的距离场 R 具有单个局部最大值或平台，所有等值线都会上升到该值，那么这些等值线可以重新路由到填充 2D 区域的单个连续螺旋路径，如图 4(c) 所示 R 。我们称这样的区域 R 为可螺旋的。不可螺旋区域有多个口袋对应于单独的局部最大值，并且不能通过所述的简单重新路由转换为单个连续螺旋。

费马螺旋线。可螺旋区域的 R 螺旋填充路径 π 可以转换为费马螺旋。如下所示，我们还可以任意选择费马螺旋遍历的起点和终点/出口点，两个点都位于区域边界上。

从点 开始 $p \in \pi$ ，在距离场上追踪向上的梯度线以 \mathcal{D}_R 与 $\mathcal{I}(p)$ 相交 π （如果存在）。如果 p 接近 \mathcal{D}_R 区域的最大值或中心 R ，路径 π 变细，则梯度线可能不相交 π 。我们将 $\mathcal{I}(p)$ 关于 π 的内部链接称为 p ；见图 5(a)。类似地，我们通过跟踪向下梯度和相交来 p 定义向外链接。 $\mathcal{O}(p)$ 如果 p 位于 ∂R ，则这样的交集将不存在。向外和向内的链接将作为转换为费马螺旋线的重新路由点。

基于向内遍历沿路径 π 施加部分顺序。因此，第一个点是区域边界上的终点 π ，最后一个点是区域中心。向内遍历 π 到达 p 前 q 两点 $p < q$ 。接下来， $\mathcal{B}(p)$ 我们施加一个离散化间距 δ ， p 并 π 用 $(\mathcal{N}(p))$ 分别 δ ，见图 5(a)。

让 p_{in} 为 π 的起点 π ，并假设我们希望费马螺旋线存在于 π 的 p_{out} 最外层部分 π 如图 5(b) 所示，我们从 p_{in} 开始，一直行进 π 直到到达 $p_1 = \mathcal{B}(p_{out})$ 。然后我们将路径向内从 π 重新路由 $p_1 = \mathcal{B}(p_{out})$ 到它的内部链接 $p_2 = \mathcal{I}(p_1)$ ，继续前进 π 直到到达 $p_3 = \mathcal{B}(\mathcal{I}(\mathcal{B}(p_2)))$ ，然后从这一点重新路由到它的内部链接。这种向内重新路由的形式被迭代执行，直到到达区域的中心，此时，遍历被反转为向外的遍历。向外重新路由是通过向外链接，从区域中心开始，通过 π 向内螺旋期间未遍历的部分，直到向外螺旋退出 p_{out} ；参见图 5(c)。

在上述转换过程中重新路由点的放置方式会导致沿着费马螺旋线的每个转弯处出现锯齿状或阶梯状。这些工件通过后期优化移除。

4 连续费马螺旋填充

在本节中，我们将描述我们的算法，用于为任意的、单连接的 2D 区域构建

连续路径填充，如连接的费马螺旋线 R 。关键是正确地重新路由从区域边界的欧几里德距离变换导出的水平集曲线或等值线 ∂R 。在口袋内，重新布线会产生费马螺旋线。在口袋和附近的分支区域之间，重新路由用于连接螺旋。

给定指定 w 等值线间距的指定路径填充宽度，我们 \mathcal{L} 使用 Clipper 算法 [Johnson 2015] 在上构建等值线集 R 。我们通过索引等值线 $c_{i,j}$ ，其中 i 表示它与区域边界的距离 ∂R ， $d(\partial R, c_{i,j}) = (i - 0.5)w$ ，并且 j 是具有相同距离索引的所有等值线中的索引 i 。例如， $c_{i,j}$ 与 $c_{i,j'}$ ， $j \neq j'$ 将属于两个不同的口袋。不失一般性，我们假设 $c_{1,1}$ 始终是外部区域边界 ∂R 。

我们构建了一棵树，称为螺旋轮廓树，其节点是等值线，其边表示它们与边权的连通性编码连接等值线的优选程度。该树用于以自下而上的方式递归地重新路由轮廓，从而产生单个连续路径。

树结构。我们首先将等高线与连续的登高值，例如 $c_{i,j}$ 和 $c_{i+1,j'}$ 连接到一个初始图中。为此，我们在 $c_{i,j}$ 上向 $c_{i+1,j'}$ 定义一个连接段为

$$\mathcal{O}_{i,j,j'} = \{\mathbf{p} \in c_{i,j} | d(\mathbf{p}, c_{i+1,j'}) < d(\mathbf{p}, c_{i+1,k}), k \neq j'\},$$

其中表示 $d(\mathbf{p}, c)$ 沿轮廓 c 从点到点的距离 \mathbf{p} 。该段 $\mathcal{O}_{i,j,j'}$ 由两个等值线之间可能的重新路由点形成。如果， $\mathcal{O}_{i,j,j'} \neq \emptyset$ 我们在 $c_{i,j}$ 和之间添加一条边 $c_{i+1,j'}$ 。分配给边缘的权重是长度（ $|\mathcal{O}_{i,j,j'}|$ ）。首选是不要在长连接段上重新布线，因为这样的段最好保持完整以形成长的、低曲率的路径。

在等高线上构建初始图后，我们计算最小权重生成树，即螺旋等高线树，以 $c_{1,1}$ 为根；见图 6(b)。树节点分为两种类型。I 类节点的度数小于或等于 2，它们对应于形成可螺旋区域的等值线。具体来说，每个这样的区域，例如 R_0, R_1, \dots, R_4 图 6(a) 中的区域，都是由类型 I 节点的路径形成的。II 型节点的度数大于 2，例如图 6(b) 中的浅蓝色节点，它们对应于分支等值线。这种等高线提供了可螺旋区域和可能的其他 II 型节点之间的界面。

改道。为了获得全局连续路径，我们以自下而上的方式重新改道等高线，从叶节点开始到根结束。有两种类型的重新路由操作。第一个将可螺旋区域中的等值线

连接起来，例如， R_0 在图 6 中，将其连接到一个起点和出口点彼此相邻的单个费马螺旋线中。此操作遵循第 3 节中描述和图 5 中所示的过程。第二个操作将费马螺旋线的起点和出口点连接到 II 型等高线，在最近的点（灰色点），如中所示-设定图。在可能的情况下，重用路由点以避免创建代表急转弯的新点。

曲线优化。到目前为止获得的刀具路径是全局连续的并覆盖输入区域 R ，但它只是 C^0 连续的并且可能存在高度不均匀的间距。在后处理中，我们对曲线进行局部优化以提高其公平性和间距。当前曲线首先根据曲率进行自适应采样，以便将更多样本放置在急转弯附近。目标函数是三项的加权和：第一项惩罚大的扰动；平滑项由弦长加权一维离散拉普拉斯算子定义；并且间距项使相邻曲线段之间的最短距离接近固定的预定义补丁间距。我们通过迭代 Gauss-Newton 求解优化，直到曲线更新变得可以忽略不计；显示优化前后路径的结果如图 7 所示。优化步骤的实现细节，包括精确的问题表述、优化过程和参数设置，可以在附录中找到。

5 结果

我们展示了具有不同凹度和空心度的形状的刀具路径生成结果。在路径连续性、急转弯数量、打印时间以及内部填充和制造表面外部的视觉质量方面，对传统之字形和等高线平行填充图案进行了比较。

3D 打印机和设置。我们的实验是在带有固件 Marlin 1.1.0-RC 的 RepRap Prusa i3 FDM 3D 打印机上进行的。打印结果和分析基于默认打印机设置，工具路径宽度设置为 0.4 毫米，层厚度设置为 0.2 毫米，最大喷嘴速度为每秒 80 毫米。G 代码用于将工具路径传输到 3D 打印机。

刀具路径生成。图 8 显示了我们的算法针对具有不同外部和内部结构的各种形状生成的工具路径。请注意，图 8 中的两个蜂窝输入形状和图 1 中的一个都是由 Lu 等人的工作构建的 3D 多孔结构的 2D 切片。[2014]。每条刀具路径都是连续的，由相连的费马螺旋线组成。所有结果均使用默认参数设置生成。初始 CFS 构造没有可调参数。对于曲线优化，参数是固定的，如附录中所述。

表 1 显示了三种填充模式的急转弯百分比和断开的刀具路径段数：传统之字形填充、等高线平行填充和我们的填充。我们不报告 CFS 的后一个数字，因为它总是产生单一路径。我们实验中显示和制作的所有锯齿形和等高线平行路径都是

使用 Slic3r 软件 [2016] 生成的。

为了计算沿刀具路径的急转弯次数，我们 π 沿每个点统一采样 50,000 个点，我们估计其积分曲率 [Pottmann 等人。 π 2009] 有一个半径为 0.2mm 的圆，这适合于制造层的大小和我们实验中的默认填充。如果一个点用于曲率估计的相关区域覆盖范围小于圆面积的 30%，则该点被视为代表急转弯。在表 1 中，我们报告了被视为急转弯的点的百分比。很明显，CFS 产生的急转弯数量远低于锯齿形，与等高线平行填充相比更接近，但通常仍低于轮廓平行填充。另一方面，后者表现出高轮廓多样性。

对于图 8 中的部分形状列表，我们在表 2 中报告了算法的运行时间；还提供了其他相关统计数据。目前，螺旋构造和连接算法在 C++中实现，而曲线优化阶段在 MATLAB 中实现。以上所有时间均在 Intel® Core™ i7-6700 CPU 4.0GHz 和 16GB RAM 上测得。

填充不足和过度填充。沿刀具路径的曲线段之间的间距不均匀会导致填充不足和过度填充。由于很难测量实际打印的这些填充伪影的范围，我们通过在其预期填充宽度处加厚计算的工具路径并在处理后测量交叉点和间隙来提供估计。图 9 可视化了一条 CFS 路径在路径优化前后的过度填充和不足填充。图 10（顶部）比较了几种形状上的填充不足和过度填充的数量。

正如人们所预料的那样，平滑会增加间隙和填充不足，尤其是在尖角附近（例如，矩形“G”的四个角和齿轮模型的许多角）。总体而言，我们的曲线优化（平滑加间隙）倾向于增加填充不足并减少过度填充；参见图 10。如图 9 所示，间隔项可以有效地消除或至少更均匀地分布未优化路径的严重过度填充。

我们当前的曲线优化方案无法填补所有空白可归因于有限的曲线移动。例如，不能拉长曲线来减轻填充不足。在尖角和转弯附近，曲线公平性和间隙大小之间存在权衡。由于差距通常很少且相距很远，因此可以在少数地方牺牲公平性来填补差距，例如，通过局部拉长曲线；我们把它留给以后的工作。

视觉质量。图 11 显示了使用三种填充图案制作的四种二维层形状（图 8 中的“S”、齿轮和两个蜂窝切片）的照片。图 10（底部）绘制了四种形状的估计填充不足和过度填充。

从视觉上看，我们观察到之字形几乎不会导致底部填充，并且通常可以沿直线刀具路径保持均匀的材料分布。但是，区域边界附近的填充质量会降低，同时显示

出粗糙度和“混叠”伪影。后者出现在接近于平行但不平行于扫描方向的边界附近（参见“S”示例）。由于锯齿形填充不是全局连续的，因此填充伪像也会出现在单独的锯齿形填充段连接的区域。就估计的过度填充而言，如图 10（左下）所示，之字形比其对应部分产生更大的量，因为 Slic3r 生成的靠近区域边界的路径的距离小于到边界的距离，导致过度填充 $\frac{1}{2}w$ -沿着这些路径填充。

对于轮廓平行填充，可见伪影（填充不足或过度填充）出现在口袋中心附近和相邻但单独轮廓区域之间。相比之下，由 CFS 产生的制造似乎表现出更好的整体质量和更少可见的伪影。但是，由于曲线平滑，CFS 似乎会导致相对大量的未填充；参见图 10（右下）。当然，应该记住，由于 3D 打印是一种物理过程，随机设备的不精确性可能会导致填充层中出现可见的伪影。

图 12 检查了使用我们的 FDM 打印机制造的 3D 对象的表面质量，对比了 CFS 填充和锯齿形填充。该物体由图 8 中齿轮层的 50 倍垂直挤压形成；最后一个圆柱体高 1 厘米。由于我们讨论过的路径平滑，从 CFS 填充的顶视图中可以看到明显的间隙。另一方面，我们的刀具路径优化有效地将（相对较大的）底部填充总量分布在大量点上，因此大多数单个间隙都很小，可以通过熔化灯丝材料来填充。从侧面看，CFS 填充导致更平滑的边界，而锯齿形填充引起的表面粗糙度是显而易见的。然而，后者通常通过外部轮廓校正，但代价是轮廓外部和内部之字形填充边界之间的底部填充。

制作时间。图 13 比较了 RepRap Prusa 3D 打印机上记录的三种填充方法的制造时间。我们观察到，虽然 CFS 工具路径的制造速度通常比 FDM 打印机上的对应工具路径更有利，但速度增益各不相同。对于更复杂的层，例如蜂窝切片，速度增益往往更显著。

与进化迷宫的比较。在图 14 中，我们将连接的费马螺线与 Pederson 和 Singh [2006] 的随机曲线演化结果进行了比较。如图所示，如果进化得到曲线与输入形状边界更好对齐的奖励，则结果在视觉上更具可比性。然而，曲线演化执行向内腐蚀，因此，不太可能像我们的螺旋方法那样保持公平且符合轮廓的外部路径。此外，局部随机运动可能会导致整个过程中出现更急转弯。

6 结论、局限性和未来工作

我们提出了一种使用连接的费马螺旋线的区域填充算法，实现了全局连续性。我们的算法将螺旋线作为空间填充曲线的使用从规则的凸形扩展到非凸形，甚至是具有许多内部孔的形状。我们的贡献是双重的。在概念层面，我们介绍了费马螺旋线的使用，以构建一种新型的空间填充模式。该结构反映了费马螺旋的引人注目的特性。使用 Fermat 螺旋线可防止曲线被锁在口袋中。此外，沿费马螺旋线边界选择起点和终点的自由度有助于系统地连接一组费马螺旋线的方案。实际上，新曲线在分层制造背景下具有用于刀具路径规划的吸引人的特性。

回想起来，连通的费马螺旋线不一定适用于所有的层形状。与他们的同行相比，他们似乎更擅长填充具有复杂几何形状(layer)，尤其是那些有很多孔的层，以实现更高的内部和外部构建质量。如果要打印具有蜂窝内部结构的 3D 物体 [Lu 等人。2014]，一个明智的计划是使用我们的 CFS 填充打印中间切片，同时打印顶部，其中层形状可能是凸面或螺旋状，具有锯齿形或混合填充，可能在它们之间交替。

如前所述，连接的费马螺旋线不能保证真正充满空间。它们也缺乏皮亚诺曲线或希尔伯特曲线所具有的规律性和数学上的严谨性；连通货马螺旋的定义是建设性的，而不是概念性的。与之字形相比，我们当前的算法通常会导致更少的急转弯。但是，它并没有尝试将它们最小化。局部曲线优化方案也留下了改进的空间。特别是，当前的曲线位移不能使相邻线段相互“滑动”或拉长曲线以填充间隙。正如 Pedersen 和 Singh [2006] 的曲线演化方案中那样，通过添加吸引力-排斥力可以实现这些操作；我们把它留给以后的工作。

我们用于分层制造的新工具路径所提供的增益量取决于 3D 打印机的电机控制器的机械结构。当代的低端打印机依赖于简单的电机控制，通过分段线性段来近似平滑曲线。人们可以将这种机制视为迎合之字形刀具路径。这也可能被认为是我们方法的局限性，因为我们寻求低曲率但非直的工具路径并且不利用这些低端打印机的控制机制。另一方面，可以结合更复杂的前瞻和自适应速度控制算法，为低曲率但非直线刀具路径实现更高的电机速度 [Wang 和 Cao 2012]。同样，具有更复杂控制器的高端打印机，如当前的计算机数字控制 (CNC) 机器，也可以为曲率较小的刀具路径实现更高的电机速度 [Wang 等人。2010]。有了这样的控制器，费马螺旋将产生显著的加速。

将来，我们想研究连续层的填充图案之间的相互作用。为了获得更好的 FDM 打印强度，不应使用接近相同的填充图案来制造连续的层。我们的构造方案可能会

略有扰动，因此连续层的费马螺旋填充可能会交织在一起。

另一个要考虑的层间优化是关于每层的起点和终点，以增加连贯性并避免喷嘴从一层到下一层的冗余移动。最后，重新检查涉及对象方向或分解的优化问题，同时考虑生成的 2D 切片的填充方式会很有趣。

附录 2 译文原文

Connected Fermat Spirals for Layered Fabrication

ABSTRACT

We develop a new kind of “space-filling” curves, connected Fermat spirals, and show their compelling properties as a tool path fill pattern for layered fabrication. Unlike classical space-filling curves such as the Peano or Hilbert curves, which constantly wind and bind to preserve locality, connected Fermat spirals are formed mostly by long, low-curvature paths. This geometric property, along with continuity, influences the quality and efficiency of layered fabrication. Given a connected 2D region, we first decompose it into a set of sub-regions, each of which can be filled with a single continuous Fermat spiral. We show that it is always possible to start and end a Fermat spiral fill at approximately the same location on the outer boundary of the filled region. This special property allows the Fermat spiral fills to be joined systematically along a graph traversal of the decomposed sub-regions. The result is a globally continuous curve. We demonstrate that printing 2D layers following tool paths as connected Fermat spirals leads to efficient and quality fabrication, compared to conventional fill patterns.

1 Introduction

The emergence of additive manufacturing technologies [Gibson et al. 2015] has led to growing interests from the computer graphics community in geometric optimization for 3D fabrication. The focus of many recent attempts has been on shape optimization: how to best configure a 3D shape, e.g., via hollowing or strengthening, to achieve quality and cost-effective fabrication. In this work, we look at the problem from a different angle. Instead of addressing the higher-level question of what to print, we examine lower-level yet fundamental issues related to how to print a given object.

At the most elementary level, additive or layered fabrication operates by moving a print head which extrudes or fuses print material layer by layer. When printing each layer, the print head follows a prescribed tool path to fill the 2D region defined by the shape of the

printed object. Topologically, continuity of a tool path is critical to fabrication. A tool path discontinuity or contour plurality forces an on-off switching of the print nozzle, negatively impacting build quality and precision [Dwivedi and Kovacevic 2004; Ding et al. 2014]. Geometrically, sharp turns and corners are undesirable since they lead to discretization artifacts at layer boundaries and cause de-acceleration of the print head, both reducing print speed and degrading fill quality [Jin et al. 2014].

Zigzag has been the most widely adopted fill pattern by today's 3D printers due to its simplicity [Gibson et al. 2015]; see Figure 2 for various fill patterns. However, a zigzag fill consists of many sharp turns, a problem that is amplified when printing shapes with complex boundaries or hollow structures. A contour-parallel tool path, formed by iso-contours of the Euclidean distance transform, provides a remedy, but it leads to high contour plurality since the iso-contours are disconnected from each other. A spiral fill pattern, for simple shapes such as a square, is continuous. However, for more complex shapes, both contour-parallel fills and spiral fills tend to leave isolated "pockets" corresponding to singularities of the distance transform, as shown in Figure 3(a). These pockets are disconnected and result in path plurality. An intriguing geometry question is whether a connected 2D region can always be filled by a continuous pattern formed by one or more spirals.

In this paper, we introduce the use of Fermat spirals [Wikipedia 2015] as a fundamental 2D fill pattern and develop a tool path planning algorithm based on connected Fermat spirals or CFS to continuously fill a connected 2D region. A Fermat spiral is an interesting space-filling pattern with two interleaving sub-spirals, one inward and one outward; see Figure 2(e). Fermat spirals had not been exploited for tool path planning before and they possess three key properties to make them attractive as a fill pattern:

1. Like contour-parallel paths, a Fermat spiral conforms to the region boundary, with one sharp turn in the center.
2. Several Fermat spirals covering a 2D region can be continuously connected. While a conventional spiral travels either inward or outward, a Fermat spiral goes in and then out, allowing several of them to be joined at their boundaries.

3. The start and exit points of a Fermat spiral can be chosen arbitrarily over its boundary; see Figures 2(e-f). This special property facilitates connections between Fermat spirals.

We develop an algorithm to construct a CFS curve to fill a singly-connected 2D region. First, the algorithm decomposes the input region into a set of sub-regions each of which admits a continuous fill by a single Fermat spiral; we call these sub-regions spirallable. The start and exit points for each Fermat spiral are placed next to each other along the spiral boundary. We then obtain a continuous traversal of the spirallable sub-regions and connect their respective Fermat spirals through the start/exit points to form a globally continuous curve. Further optimization can be applied to improve fairness and spacing. The resulting curve has fewer sharp turns than a zigzag fill and composed mostly of long, low-curvature paths.

We show CFS curves constructed for input shapes with varying exterior and interior complexity. We fabricate 2D layers and 3D prints using CFS patterns and compare the results visually and numerically to those provided by conventional fills via zigzag and contour-parallel tool paths. The new fill pattern appears to excel at handling shapes with much concavity and many interior holes.

In retrospect, the desirable properties we seek from connected Fermat spirals are almost completely opposite to those possessed by classic space-filling curves such as Hilbert curves; see Figure 2(c). Hilbert or Peano curves are designed to wind and bend to preserve locality of the space traversal. CFS curves are meant to avoid bending as much as possible to attain a higher degree of fairness. In general, CFS curves are not guaranteed to completely cover an arbitrary 2D region even at infinitely high resolution. As well, our new curves do not possess recursive properties as the classical space-filling curves. For layered fabrication however, the new curve is clearly more attractive than fractal-like fill patterns.

2 Background and Related Work

Recently, there has been a flourishing of works in computer graphics on optimizing 3D shapes or their configurations for efficient and effective fabrication, e.g., to ensure or

improve physical stability [Pre'vost et al. 2013], structural strength [Stava et al. 2012; Hildebrand et al. 2013; Lu et al. 2014], or appearance [Zhang et al. 2015] of the print, to save material [Vanek et al. 2014b; Hu et al. 2014], and to adapt to the limited print volume [Luo et al. 2012; Vanek et al. 2014a; Chen et al. 2015; Yao et al. 2015]

In this section, we focus on the tool path planning problem in the context of additive manufacturing (AM). First, to explain the importance of continuity and fairness for the tool paths, we present some background material on nozzle mechanisms that control the viscoelastic material that is extruded along the path, as well as the mechanics of the motors that control the tool paths. Then we discuss related geometric and engineering approaches for optimizing tool paths. Our coverage is not meant to be exhaustive. Interested readers should refer to the book by Gibson et al. [2015], short surveys from [Kulkarni et al. 2000; Ding et al. 2014], as well as the recent SIGGRAPH course by Dinh et al. [2015].

Tool path continuity. Fused Deposition Modeling or FDM is the most widely applied AM technology. During the FDM process, filament is melted into viscoelastic material and extruded from a small opening of the print nozzle. Due to liquid compressibility, it is generally hard to predict the amount of viscoelastic material to emit in order to create a continuous extrusion control. Consequently, the first portion of the filament from the nozzle usually under- or over-fills. Such uneven fills cause visual artifacts when they occur near the surfaces of the printed object. When they occur between fill paths, attachment between filaments can be weakened, lowering the strength of the print. When turning off the extrusion, the temporal gap between the stopping of the feed motor and that of the filament extrusion is difficult to control, which again leads uneven fills. Similar situations also arise when the print material is fused by the print head, e.g., for powder-based printing. As well, any discontinuity along a tool path necessitates a nozzle movement which does not contribute to the print. Thus, the primary objective in designing tool paths is to minimize the on/off switching along the path, or in other words, to maximize its continuity.

Tool path geometry. The geometry of tool paths, in particular their curvature, influences fabrication time and quality. As a tool path rounds itself about a sharp turn,

more de-acceleration and acceleration times are required, causing more of a slow-down of the extrusion head, as compared to the case of a soft turn. As well, acute turn angles also lead to more over-fill or under-fill of the filament [Jin et al. 2014]. Hence, a long and continuous tool path without sharp turns may enable the extrusion head to move along the whole tool path at a speed that is close to the highest allowed with small changes, leading to efficient and quality fabrication.

Direction-parallel vs. contour-parallel fills. The most popular fill method in commercial AM systems follows the zigzagging pattern [Ding et al. 2014]. Along with raster scans, zigzagging belongs to the class of direction-parallel fills. In contrast, contour-parallel paths are comprised of a set of closed contours parallel to the outline of a given slice [Yang et al. 2002]. Over simple 2D regions, such paths lead to smoother turns and object boundaries compared zigzagging [El-Midany et al. 1993], but they always have a high contour plurality. Hybrid fills have also been proposed [Jin et al. 2013]; they generate a few contours inward before filling the remaining interior area with a zigzag, but attachment between the two fill patterns can become suspect. When the 2D slice to be filled has a complex shape with many concavities, standard implementations of both fill patterns are prone to discontinuity issues.

Spiral tool paths. Spiral tool paths have been widely applied for pocket machining [Ren et al. 2009]. Held and Spielberger [2014] decompose a 2D layer into spirallable pockets and machine each pocket following a separate, classical spiral pattern; no globally continuous path was constructed. Spiral tool paths are less common for AM and one major reason (also applicable to contour-parallel fills) is that due to a lack of direction bias, spiral patterns for adjacent slices replicate each other and cannot be “cross-weaved” at an angle; this could compromise fabrication strength for FDM printers [Gibson et al. 2015]. This problem may be fixed by hybrid fills, e.g., alternating between spiral and zigzagging layers. Our work focuses on how to optimize the continuity of spiral fills.

Space-fill curves. A continuous tool path that fills a 2D region is a space-filling curve (SFC). SFCs have been adopted for various applications, e.g., image encoding [Dafner et al. 2000] and maze design [Pedersen and Singh 2006]. Fractal-like SFCs have been suggested as fill patterns for AM [Wasser et al. 1999]. However, they are complex

to realize and full of sharp turns. The tool path fill problem has some resemblance to lawn mowing [Arkina et al.2000], which is formulated under a rather different setting: it seeks the shortest path for a “cutter” with a prescribed shape to cover all points (possibly multiple times) in a 2D region.

Labyrinths. Mazes and many famous labyrinth patterns are also space-filling [Wikipedia 2016]. A unicursal labyrinth curve is continuous and starts and ends at the same point, just like our connected Fermat spirals. Pedersen and Singh [2006] developed a stochastic curve evolution algorithm to produce expressive, space-filling labyrinths where the Brownian motion of the curve particles are subject to attraction-repulsion forces, as well as local fairness and field alignment constraints. In contrast, our algorithm is top-down and with more global structural control in the construction. The resulting curves have smoother boundaries and less turns.

Domain decomposition. One interesting way to obtain a continuous tool path is to decompose a 2D region into several sub-regions each of which admits a continuous fill. Then these regional fills are connected to achieve global continuity. Along these lines, [Dwivedi and Kovacevic 2004] decompose a polygon into monotone sub-polygons and fill each sub-polygon using a closed zigzagging curve, along which the start/entry point can be chosen arbitrarily. Ding et al. [2014] execute convex decomposition and for each convex polygon, an optimal zigzagging direction is found to facilitate continuous connection between the polygon fills. However, both methods were designed to deal with polygon inputs and cannot properly handle shapes with smooth concave boundaries.

Our work also relies on a region decomposition while it can deal with arbitrary 2D shapes as input. Instead of using zigzags, we employ Fermat spiral fills to achieve both continuity and a higher degree of fairness. The decomposition scheme is designed to accommodate contour-parallel and spiral tool paths.

Continuously fillable shapes. Polygon convexity [Ding et al.2014] and monotonicity [Dwivedi and Kovacevic 2004] were chosen in the domain decomposition approaches to achieve tool path continuity since both shape properties ensure a continuous fill by the zigzagging pattern. For monotone polygons, a limited set of scan directions guarantee this, while for a convex polygon, any scan direction leads to a continuous zigzagging fill.

Spirallability is their counterpart for spiral or contour-parallel fills and to the best of our knowledge, such a shape property has not been studied before.

3 Spirals, Fermat Spirals, and Spirallability

Before introducing spirals, we first study contour-parallel tool paths and relate them to the Euclidean distance transform of a 2D region. Then, we describe how to convert the parallel contours into a regularly spaced spiral pattern and define spirallability. Finally, a spiral fill is converted into a Fermat spiral fill, where we can choose the start and end points on the boundary arbitrarily.

Contour-parallel path as iso-contour. Let R be a connected 2D region whose boundary is denoted by ∂R . A Euclidean distance transform for ∂R defines a scalar distance field \mathcal{D}_R over R where for each point $p \in R$, $\mathcal{D}(p)$ is the shortest distance from p to ∂R . An iso-contour associated with distance d is composed of all points in R whose scalar value is d ; the boundary ∂R is the iso-contour associated with the iso-value 0. Subject to the width of the fill material for the fabrication process, the set of contour-parallel tool paths correspond to a set of equidistant iso-contours, which are all disconnected from each other, as shown in Figure 4(a).

Spiral and spirallability. Two adjacent iso-contours can be connected to form a single continuous path by breaking and rerouting the contours, as shown in Figure 4(b). Rerouting adjacent contours in such an offsetting fashion would lead to a spiral pattern. If the distance field within R has a single local maximum or plateau, to which all the iso-contours would ascend, then these contours can be rerouted into a single continuous spiral path that fills the 2D region R , as shown in Figure 4(c). We call such a region R spirallable. Non-spirallable regions have multiple pockets corresponding to separate local maxima and cannot be converted into a single continuous spiral with a simple rerouting as described.

Fermat spiral. A spiral fill path π for a spirallable region R can be converted into a Fermat spiral. As we show below, we can also choose the start and end/exit points of the Fermat spiral traversal arbitrarily, with both points lying on the region boundary.

Starting from a point $p \in \pi$, trace the upward gradient line over the distance field

\mathcal{D}_R to intersect π at $\mathcal{I}(p)$, if it exists. If p is close to the maximum of \mathcal{D}_R or center of the region R , where the path π is thinning out, then the gradient line may not intersect π . We call $\mathcal{I}(p)$ the inward link for p with respect to π ; see Figure 5(a). Similarly, we define the outward link $\mathcal{O}(p)$ for p by tracing the downward gradient and intersect. If p lies on ∂R , then such an intersection would not exist. The outward and inward links will serve as rerouting points for the conversion to a Fermat spiral.

To help describe the rerouting procedure, we impose a partial order $<$ along path π based on inward traversal. Thus, the first point is the end point of π on the region boundary and the last point is at the region center. Two points $p < q$ if the inward traversal along π reaches p before q . Next, we impose a discretization spacing δ and denote the point preceding (respectively, succeeding) p along π at a distance δ by $\mathcal{B}(p)$ (respectively, $\mathcal{N}(p)$); see Figure 5(a).

Let p_{in} be the starting point of π and suppose that we would like the Fermat spiral to exist at p_{out} along the outermost portion of π . As shown in Figure 5(b), we start at p_{in} and travel along π until reaching $p_1 = \mathcal{B}(p_{out})$. Then we reroute the path inward from $p_1 = \mathcal{B}(p_{out})$ to its inward link $p_2 = \mathcal{I}(p_1)$, continue traveling along π until reaching $p_3 = \mathcal{B}(\mathcal{I}(\mathcal{B}(p_2)))$, and reroute from this point to its inward link. This form of inward rerouting is executed iteratively until reaching the center of the region, at which point, the traversal is reversed into an outward one with a turn. The outward rerouting is through the outward links, starting at the region center, passing through portions of π that were not traversed during the inward spiral, until the outward spiral exits at p_{out} ; see Figure 5(c).

The way the rerouting points are placed in the above conversion procedure leads to jaggies or staircasing at every turn along the Fermat spiral. These artifacts are removed by a post-optimization.

4 Continuous Fermat Spiral Fill

In this section, we describe our algorithm for constructing a continuous path fill, as connected Fermat spirals, for an arbitrary, singly-connected 2D region R . The key is to properly reroute levelset curves or iso-contours derived from the Euclidean distance

transform of the region boundary ∂R . Within a pocket, the rerouting produces a Fermat spiral. Between pockets and near branching regions, rerouting serves to connect the spirals.

Given a prescribed path fill width w specifying spacing between iso-contours, we construct the set \mathcal{L} of iso-contours using the Clipper algorithm [Johnson 2015] over R . We index an iso-contour by $c_{i,j}$, where i indicates its distance from the region boundary ∂R , $d(\partial R, c_{i,j}) = (i - 0.5)w$, and j is an index among all iso-contours with the same distance index i . For example, $c_{i,j}$ and $c_{i,j'}$ with $j \neq j'$, would belong to two separate pockets. Without loss of generality, we assume that $c_{1,1}$ is always the outer region boundary ∂R .

We build a tree, called the spiral-contour tree, whose nodes are the iso-contours and whose edges denote their connectivity with edge weights encoding how preferable it is to connect the iso-contours. The tree is used to recursively reroute the contours in a bottom-up fashion, producing a single continuous path.

Tree construction. We first connect iso-contours with consecutive iso-values, e.g., $c_{i,j}$ with $c_{i+1,j'}$, into an initial graph. To this end, we define a connecting segment on $c_{i,j}$ towards $c_{i+1,j'}$ as

$$\mathcal{O}_{i,j,j'} = \{\mathbf{p} \in c_{i,j} | d(\mathbf{p}, c_{i+1,j'}) < d(\mathbf{p}, c_{i+1,k}), k \neq j'\},$$

where $d(\mathbf{p}, c)$ denotes the distance from a point \mathbf{p} to points along a contour c . The segment $\mathcal{O}_{i,j,j'}$ is formed by possible rerouting points between the two iso-contours. We add an edge between $c_{i,j}$ and $c_{i+1,j'}$ to the graph if $\mathcal{O}_{i,j,j'} \neq \emptyset$. The weight assigned to the edge is $\text{length}(\mathcal{O}_{i,j,j'})$. The preference is to not reroute over a long connecting segment since such a segment is preferred to remain intact to form long, low-curvature paths.

After building the initial graph on iso-contours, we compute a minimum-weight spanning tree, the spiral-contour tree, with $c_{1,1}$ as the root; see Figure 6(b). The tree nodes fall into two types. Type I nodes have degrees less than or equal to two and they correspond to iso-contours that form spirallable regions. Specifically, each such region, e.g., R_0, R_1, \dots, R_4 in Figure 6(a), is formed by a path of Type I nodes. Type II nodes

have degrees greater than two, e.g., those colored in light blue in Figure 6(b), and they correspond to branching iso-contours. Such an iso-contour provides an interface between spirallable regions and possibly other Type II nodes.

Rerouting. To obtain a globally continuous path, we reroute the iso-contours in a bottom-up fashion, starting from leaf nodes and ending at the root. There are two types of rerouting operations. The first connects iso-contours in a spirallable region, e.g., R_0 in Figure 6, into a single Fermat spiral with start and exit points next to each other. This operation follows the procedure described in Section 3 and illustrated in Figure 5. The second operation connects the start and exit points of a Fermat spiral to a Type II iso-contour, at the closest points (gray points), as shown in the in-set figure. Wherever possible, rerouting points are reused to avoid creating new points representing sharp turns.

Curve optimization. The tool path obtained so far is globally continuous and covers the input region R , but it is only C^0 continuous and possibly suffers from highly nonuniform spacing. In post-processing, we locally optimize the curve to improve its fairness and spacing. The current curve is first adaptively sampled based on curvature so that more samples are placed near sharp turns. The objective function is a weighted sum of three terms: the first term penalizes large perturbations; the smoothing term is defined by a chord-length weighted 1D discrete Laplacian; and the spacing term keeps shortest distances between adjacent curve segments close to a fixed, pre-defined patch spacing. We solve the optimization via iterative Gauss-Newton until curve updates become negligible; a result showing the paths before and after optimization is shown in Figure 7. Implementation details of the optimization step, including precise problem formulation, optimization procedure, and parameter setting, can be found in the appendix.

5 Results

We show tool path generation results on shapes with varying degrees of concavity and hollowness. Comparisons are made to conventional zigzag and contour-parallel fill patterns in terms of path continuity, amount of sharp turns, print time, as well as visual quality of the interior fill and fabricated surface exterior.

3D printer and setting. Our experiments have been conducted on a RepRap Prusa i3

FDM 3D printer with firmware Marlin 1.1.0-RC. Printing results and analyses are based on the default printer setting, with tool path width set at 0.4mm, layer thickness at 0.2mm, and maximal nozzle speed at 80 mm per second. G-code is used to transfer the tool paths to the 3D printer.

Tool path generation. Figure 8 shows tool paths generated by our algorithm for a variety of shapes with varying exterior and interior structures. Note that the two honeycomb input shapes in Figure 8 and the one from Figure 1 are all 2D slices of the 3D porous structures constructed by the work of Lu et al. [2014]. Each tool path is continuous and composed of connected Fermat spirals. All results are produced with the default parameter setting. There are no tunable parameters for initial CFS construction. For curve optimization, the parameters are fixed as discussed in the Appendix.

Table 1 shows the percentage of sharp turns and the number of disconnected tool path segments for three fill patterns: conventional zigzag, contour-parallel fills, and ours. We do not report the latter number for CFS since it always produces a single path. All the zigzag and contour-parallel paths shown and fabricated in our experiments were generated with the Slic3r software [2016].

To count the number of sharp turns along a tool path π we uniformly sample 50,000 points along π and at each point, we estimate its integral curvature [Pottmann et al. 2009] with a circle of radius 0.2mm, which is appropriate for the size of fabricated layers and the default fill with in our experiments. A point is deemed to represent a sharp turn if the smaller of its associated area coverage for curvature estimation is less than 30% of the circle area. In Table 1, we report the percentages of points deemed as sharp turns. It is quite evident that the number of sharp turns produced by CFS is much lower than that of zigzags and it is more comparable to, but generally still lower than, that of contour-parallel fills. On the other hand, the latter exhibits high contour plurality.

We report running times of our algorithm in Table 2 for a partial list of shapes in Figure 8; other relevant statistics are provided as well. Currently, the spiral construction and connection algorithm is implemented in C++ while the curve optimization phase is implemented in MATLAB. All of the above times are measured on an Intel® Core™ i7-6700 CPU 4.0GHz with 16GB RAM.

Under- and over-fill. Under- and over-fills occur as a result of non-uniform spacing between curve segments along a tool path. Since it is difficult to measure the extent of these fill artifacts for real prints, we provide an estimate by thickening a computed tool path at its expected fill width and measure the intersection and gap after the process. Figure 9 visualizes the over- and under-fills for one CFS path before and after path optimization. Figure 10 (top) compares the amount of under- and over-fills over several shapes.

As one would expect, smoothing tends to increase gaps and under-fills, especially near sharp corners (e.g., four corners of the rectangular ‘G’ and many corners of the gear model). Overall, our curve optimization (smoothing plus spacing) tends to increase under-fills and reduce over-fills; see Figure 10. As shown in Figure 9, the spacing term is seen to effectively remove or at least, more evenly distribute, severe over-fills of an un-optimized path.

The inability of our current curve optimization scheme to fill all the gaps can be attributed to limited curve movements. For example, curves cannot be elongated to alleviate under-fills. Near sharp corners and turns, there is a trade-off between curve fairness and gap size. Since the gaps are typically few and far inbetween, sacrificing fairness at few places to fill the gaps, e.g., by elongating the curve locally, is possible; we leave this for future work.

Visual quality. Figure 11 shows photos taken of four 2D layer shapes (the ‘S’, gear, and the two honeycomb slices from Figure 8) fabricated using the three fill patterns. Figure 10 (bottom) plots the estimated under- and over-fills over the four shapes.

Visually, we observe that zigzag incurs little under-fill and can generally maintain even material distribution along straight tool paths. However, fill quality degrades near region boundaries, showing both roughness and “aliasing” artifacts. The latter shows up near boundaries which are close to being parallel to, but are not parallel to, the scan direction (see the ‘S’ example). Since the zigzag fill is not globally continuous, fill artifacts also occur over areas where separate zigzag-filled segments join. In terms of estimated over-fills, as shown in Figure 10 (bottom left), zigzag incurs a larger amount than its counterparts since paths generated by Slic3r next to region boundaries have a

distance smaller than $\frac{1}{2}w$ to the boundaries, leading to excessive over-fills along these paths.

For contour-parallel fills, visible artifacts (under- or over-fills) occur near the center of pockets and between adjacent, but separately contoured regions. In contrast, fabrication resulting from CFS appear to exhibit better overall quality with less visible artifacts. However, CFS appears to lead to relatively large amount of under-fills due to curve smoothing; see Figure 10 (bottom right). Of course, one should bear in mind that since 3D printing is a physical process, random device imprecisions which may cause visible artifacts in the filled layers are possible.

Figure 12 examines the surface quality of a 3D object fabricated using our FDM printer, contrasting CFS fills to zigzag fills. The object is formed by a 50-fold vertical extrusion of the gear layer from Figure 8; the final cylinder is 1cm tall. There are visible gaps from a top view of the CFS fills, due to path smoothing as we discussed. On the other hand, our tool path optimization effectively distributes the (relatively large) total amount of under-fills over a large number of spots so that most individual gaps are small and can be filled by melting of the filament material. From the side views, CFS fills lead to smoother boundaries, while surface roughness arising from zigzag fills is evident. However, the latter is typically corrected by external contouring, but at the expense of under-fills between the contoured exterior and boundaries of the interior zigzag fills.

Fabrication time. Figure 13 compares fabrication times recorded on the RepRap Prusa 3D printer for the three fill methods. We observe that while the fabrication speed for CFS tool paths is generally more favorable than their counterparts on an FDM printer, the speed gains vary. For more complex layers, e.g., the honeycomb slices, the speed gains tend to be more significant.

Comparison to evolved labyrinths. In Figure 14, we compare our connected Fermat spirals to a result from stochastic curve evolution by Pederson and Singh [2006]. The results become more visually comparable if the evolution is rewarded by better alignment of the curves with the boundary of the input shape, as shown. However, the curve evolution performs inward erosions and as such, it is unlikely to maintain a fair and

outline-conforming exterior path as our spiral approach. Moreover, the local stochastic movements are likely to result in more sharp turns throughout.

6 Conclusion, limitation, and future work

We present a region fill algorithm using connected Fermat spirals, achieving global continuity. Our algorithm extends the use of spirals as space-filling curves from regular convex shapes to non-convex shapes, even shapes with many interior holes. Our contributions are two-fold. At a conceptual level, we introduce the use of Fermat spirals to the construction of a new kind of space-filling patterns. The construction reflects compelling properties of Fermat spirals. The use of Fermat spirals prevents the curve from being locked in pockets. Furthermore, the freedom allowed in choosing start and end points along the boundary of a Fermat spiral facilitates a scheme which systematically joins a set of Fermat spirals. Practically, the new curves possess appealing properties for tool path planning in the context of layered fabrication.

In retrospect, connected Fermat spirals are not necessarily suitable for all layer shapes. Compared to their counterparts, it appears that they excel at filling layers with complex geometry, especially those with many holes, to achieve higher build quality both inside and on the exterior. If one were to print a 3D object with honeycomb interiors [Lu et al. 2014], a sensible plan would be to print the middle slices using our CFS fills while topping off the print, where the layer shapes are likely to be convex or spirallable, with zigzag or hybrid fills, possibly alternating between them.

As discussed previously, the connected Fermat spirals are not guaranteed to be truly space-filling. They also lack the regularities and mathematical rigors possessed by Peano or Hilbert curves; the definition of connected Fermat spirals is constructive and not conceptual. Our current algorithm generally results in less number of sharp turns compared to zigzag. However, it makes no attempt to minimize them. The local curve optimization scheme also leaves room for improvements. In particular, current curve displacements cannot “slide” adjacent segments against each other or elongate the curve to fill gaps. These operations are possible by adding attraction-repulsion forces as in the curve evolution scheme of Pedersen and Singh [2006]; we leave this for future work.

The amount of gains afforded by our new tool paths for layered fabrication is dictated by the mechanics of the motor controllers of the 3D printers. Contemporary, low-end printers rely on simple motor controls, approximating a smooth curve by piecewise linear segments. One may regard such a mechanism as catering to zigzagging tool paths. This may also be accounted for as a limitation of our approach, as we seek low-curvature but non-straight tool paths and do not take advantage of the control mechanisms of these low-end printers. On the other hand, it is possible to incorporate more sophisticated look-ahead and adaptive speed control algorithms to achieve higher motor speed for low-curvature but non-straight tool paths [Wang and Cao 2012]. As well, higher-end printers with more sophisticated controllers, like those of current Computer Numerical Control (CNC) machines, can also achieve a higher motor speed for tool path with smaller curvatures [Wang et al. 2010]. With such controllers, Fermat spirals would incur a significant speed-up.

In the future, we would like to investigate the interplay between fill patterns of consecutive layers. For better strength of FDM prints, consecutive layers should not be fabricated with close-to-identical fill patterns. Our construction scheme may be slightly perturbed so that the Fermat spiral fills of consecutive layers may interweave.

Another inter-layer optimization to consider is with respect to the start and end points of each layer, in term of increasing the coherence and avoiding redundant moves of the nozzle from one layer to the next. Finally, it would be interesting to re-examine optimization problems involving object orientation or decomposition while taking into account how the resulting 2D slices are filled.