



面向非圆曲线的CAM软件开发

|  |  |
| --- | --- |
| **学 院：** | **机电工程学院** |
| **专 业：** | **智能制造工程** |
| **年级班别：** | **21级智能制造工程2班** |
| **学生姓名：** | **陈志荣 范思齐 沈家杭** |
| **学生学号：** | **3121000034 3121000036 3121000048** |
| **指导教师：** | **姜永军 杨** |

**2024年5月**

广东工业大学“数控技术”课程设计任务书

|  |  |
| --- | --- |
| 题目名称 | 面向非圆曲线的CAM软件开发 |
| 学生学院 | 机电工程学院 |
| 专业班级 | 机电/智能制造21 |
| 姓 名 | 范思齐 陈志荣 沈家杭 |
| 学 号 | 3121000035 3121000034 3121000048 |

一、课程设计的内容

用计算机高级编程语言（如VB，VC++，C#，Python等）来实现非圆曲线的计算机辅助制造(CAM)软件的开发，针对不同的非圆曲线，可任选（1）直线逼近（如等间距法、等弦长法、等误差法等）、或（2）圆弧逼近的方法产生节点。要求在满足允许误差的前提下，使得逼近的直线段或圆弧段的数量最少（即最优解）， 根据加工曲线轮廓自动生成刀具中心轨迹，自动生成加工NC代码，并能模拟实际加工走刀过程。

二、课程设计的要求与数据

**针对非圆曲线的逼近处理，要求：**

（1）列出一般的直线或圆弧逼近的算法（流程图）。

（2）列出改进的直线或圆弧逼近的算法（流程图）——即优化算法。比较改进前与改进后的两种算法结果。

**刀具补偿功能的实现：**

（1）针对给定的某一由非圆曲线所构成的平面轮廓，根据指定的走刀方向、起刀点，自动生成CNC代码。

(2)有刀具自动补偿功能（左右刀补均要有实现），根据给定的补偿量和进给方向自动计算刀具中心轨迹，有过切报警功能。

**对逼近处理后的轮廓进行仿真加工，要求：**

（1）列出要采用的插补算法的四个象限的流程图，设置进给轴的脉冲当量以及插补运算周期，分别实现脉冲增量插补和数字增量插补算法，对比两个插补算法的不同。

（2）在屏幕上显示该非圆曲线所构成的平面轮廓。根据给定的进给速度能模拟加工过程，并在屏幕上留下刀具所走中心轨迹。

**非圆曲线选择：**

0：抛物线

1: 椭圆曲线

2: 阿基米德螺线/等速螺线（900︒）

3：幂**函数曲线。**f(x)=xa的函数，a=4或8

4：幂**函数曲线。**f(x)=xa的函数，a=1/4或1/8

5：焦点Y轴上的双曲线

6：心脏线

7：匀加减速盘形凸轮廓线

8：余弦加减速盘形凸轮廓线（四段：升程-远休-回程-近休）

9：正弦加减速盘形凸轮廓线（四段：升程-远休-回程-近休）

三、课程设计应完成的工作

每组学生应在规定时间内，独立完成所选题目。运用VC或其它编程语言，编写计算机软件在WINDOWS实现数控装置的计算机仿真。要求清楚地分析问题、提出算法、确定人机界面、列出流程图，最后用程序验证，完成软件测试，并且提交程序说明书。

要求用编写计算机软件的方法解决典型非圆曲线的CAM问题。可以任选用自己熟悉的一种编程语言，要求清楚地分析问题、提出算法、列出流程图，最后用程序验证，并且提交程序说明书。

四、课程设计进程安排

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 设计各阶段内容 | 地点 | 起止日期 |
| 1 | 1、布置任务，领取课程设计任务书，了解课程设计的目的、内容和要求；了解课程设计的步骤；  2、理解本课程设计题目的具体内容要求，根据各自不同情况选择题目； | 学生宿舍 | 13周  周一~周二 |
| 2 | 1：了解和掌握有关软件开发的知识，如VB编程、VC编程、软件工程、软件开发的常用技巧及注意事项；  2：调查研究，收集资料，查阅文献。学生对所选题目进行算法设计及方案设计。 |  | 13周  周三~周五 |
| 3 | 1：掌握数控结构设计要求，具体技术指标和计算要求; 进行软件设计；  2: 进行数控系统和算法的软件编程与开发，初步实现系统的基本功能 | 学生宿舍 | 14周  周一~周二 |
| 4 | 1：通过多个实例来验证和改进系统功能，完善软件界面  2：对所开发的软件程序进行标识和说明  3：按要求的格式编制课程设计说明书  4：课程设计答辩 | 学生宿舍 | 14周  周三~周五 |

五、应收集的资料及主要参考文献

1廖效果.数控技术. 武汉:湖北科学技术出版社.2000.7

2刘又午.数字控制机床.北京:机械工业出版社

3杨林,李继良. Visual Basic编程高手.北京:北京大学出版社.2000

4一组专用凸轮的计算机辅助设计.机械工程师1998,(4):p58-59

5平面凸轮机构CAD系统的研究与开发.机械设计与制造2000,(5):p12-13

6圆柱非圆曲线槽凸轮的数控加工.制造技术与机床2000,(8):p34

7圆柱凸轮的参数化设计及数控加工.精密制造及自动化2001,11:p28

8参数化凸轮轮廓转换及NC代码自动生成.机床与液压2001,6,p29~31

9非圆曲线的等误差拟合数控节点算法研究. 数字技术与应用.2010.07

10[椭圆弧等误差直线拟合的算法研究与轨迹仿真](http://www.cnki.com.cn/Article/CJFDTOTAL-SDGC201001026.htm)[J].山东理工大学学报(自然科学版) 2010.01

11赵佩凤.刀具补偿控制系统的研究与开发[D].大连交通大学.2010

12孙海洋,范大鹏.一种新的刀具圆弧过渡补偿算法[J].中国机械工程.2007.03

发出任务书日期：2024 年 4 月 24 日

计划完成日期：2024 年 5 月 31 日

指导教师签名：

基层教学单位责任人签章：

主管院长签章：

**摘要**

本次数控课程设计我们基于QT界面开发平台，采用C++实现了非圆曲线的CAM软件的开发。我们根据任务书选择了指数小于1的幂函数曲线、余弦加减速盘形凸轮廓线（包括升程、远休、回程、近休四个阶段）两种曲线，基于等间距插补算法、等弦长插补算法、等误差插补算法及其优化算法，求解出各个算法下两种曲线的节点并连成轮廓曲线，再根据刀具补偿基本原理，根据此生成的轮廓生成左、右刀补后的刀具中心轨迹节点并连成刀具走过的路径曲线，并在QT的ui界面中的画布中进行显示。

本项目人机界面友好，较好地实现了项目预期功能，具体包括坐标系选择、对刀点的确立、运动参数与曲线参数设置、算法比较界面、NC代码的生成与保存、模拟加工暂停与停止、过切报警等功能。

**关键词：**数控加工，非圆曲线插补，自动编程，直线逼近

**Abstract**

For this CNC course project, we developed CAM software for non-circular curves using C++ on the QT interface development platform. Following the task specifications, we chose power function curves with an exponent less than 1 and cosine acceleration/deceleration disc cam profiles, encompassing lift, far rest, return, and near rest phases. Utilizing the equal spacing, equal chord length, and equal error interpolation algorithms, along with their optimized versions, we determined the nodes for each curve and assembled them into contour curves. Subsequently, applying the fundamental principles of tool compensation, we generated the left and right tool center trajectory nodes from the generated contour, connecting them to depict the tool's path, which is displayed within a zoom-enabled canvas on the QT UI interface.

The project boasts a user-friendly interface and effectively fulfills its intended functions, including coordinate system selection, tool setting, motion and curve parameter configuration, algorithm comparison interface, NC code generation and saving, machining simulation pause and termination, and overcut alert features.

**Key words**: CNC machining, non-circular curve interpolation, automated programming, linear approximation

目录

[摘要 VI](#_Toc21258)

[Abstract VII](#_Toc26055)

[目录 VIII](#_Toc11367)

[1 引言 1](#_Toc22704)

[1.1 概述 1](#_Toc7495)

[1.2 使用软件及语言 1](#_Toc21850)

[1.2.1 Qt简介 1](#_Toc13579)

[1.2.2 C++简介 2](#_Toc29164)

[2 非圆曲线的逼近算法与算法设计 4](#_Toc3163)

[2.1 非圆曲线插补算法的原理与实现 4](#_Toc23302)

[2.1.1 非圆曲线的插补算法 4](#_Toc8588)

[2.1.2 等间距直线逼近插补 4](#_Toc20759)

[2.1.3 等弦长直线逼近插补 6](#_Toc10515)

[2.1.4 等误差直线逼近插补 7](#_Toc23925)

[2.2 幂函数（指数a∈（0，1））曲线的逼近算法 9](#_Toc31422)

[2.2.1 等间距直线逼近算法 9](#_Toc25922)

[2.2.2 等误差直线逼近算法 12](#_Toc18828)

[2.3 余弦加减速盘形凸轮轮廓的逼近算法 15](#_Toc12380)

[2.3.1 等间距直线逼近算法 17](#_Toc851)

[2.3.2 等误差直线逼近算法 19](#_Toc27883)

[3 实现的功能 21](#_Toc24219)

[3.1 刀补功能 21](#_Toc15440)

[3.2 防过切功能 23](#_Toc4185)

[3.3 NC代码生成和保存功能 24](#_Toc31861)

[3.4 模拟加工功能 26](#_Toc12395)

[3.5 绘图区域缩放 26](#_Toc22746)

[4 软件界面 27](#_Toc19204)

[4.1 主界面总体介绍 27](#_Toc18645)

[4.2 路径参数设置区 28](#_Toc20072)

[4.2.1 限定起终点大小 29](#_Toc3277)

[4.2.2 限定幂数大小 29](#_Toc16302)

[4.2.3 限制角度 30](#_Toc2701)

[4.3 插补参数设置区 30](#_Toc13916)

[4.3.1 刀具半径输入优化 31](#_Toc10455)

[4.3.2 逼近方法及其参数设置逻辑 32](#_Toc24092)

[4.4 曲线数据显示区 33](#_Toc3294)

[4.5 机床参数设置区 35](#_Toc30424)

[4.6 小组成员区 36](#_Toc28759)

[4.7 NC代码操作区 37](#_Toc20227)

[4.7.1 NC代码生成 37](#_Toc15673)

[4.7.2 NC文件保存 46](#_Toc20694)

[4.8 绘图区 50](#_Toc17518)

[4.8.1 窗口创建和坐标轴绘制 51](#_Toc17098)

[4.8.2 绘图区域缩放 51](#_Toc20216)

[4.8.3 主绘图函数 52](#_Toc7324)

[4.9 主操作按钮区 53](#_Toc1369)

[4.9.1 “生成/更新原始曲线”按钮 53](#_Toc7570)

[4.9.2 “生成/更新刀具路径”按钮 54](#_Toc18251)

[4.9.3 “进行加工仿真”按钮 55](#_Toc30180)

[4.10 加工进度条 55](#_Toc7678)

[4.11 UI设计补充 56](#_Toc9332)

[4.12 源码发布 57](#_Toc1397)

[5 总结 58](#_Toc29497)

# 引言

## 概述

本次课程设计的题目是面向非圆曲线的 CAM 软件开发，我们组将以**幂函数曲线**和**余弦加减速盘形凸轮轮廓**为内容进行 CAM 软件开发，并实现相关功能。在逼近算法方面，二者的一般算法均为**等间距直线逼近**，**优化算法均为等误差直线逼近**，通过比较两种方法的插补段数进而形成对比。前期通过查找相关文献资料，对相关曲线的参数进行细致了解，然后设计二者的逼近算法，最后我们小组选择使用 **C++**和 **QT** 实现对算法的实现以及人机界面的开发。

实现功能：

1. **多种曲线插补及算法优化**。用户能够通过输入对应曲线的参数利用直线插补（等间距、等弦长、等误差）来模拟插补过程；
2. **设置左右刀补**。用户可根据实际加工需求设置左右刀补加工轨迹；
3. **图像输出及加工模拟**。同时能输出图形实际图像、刀具轨迹以及模拟刀具实际的运行；
4. **机床参数设定**。如：坐标系的选取、设定对刀点、主轴转速、主轴转向、进给速度、刀具半径、加工厚度；
5. **界面调节**。用户可以设定软件文字大小以及图形界面大小、细节缩放；
6. **具备NC代码输出功能**。能根据设定参数输出NC代码，并能读取当前输出NC代码的时间，同时也能将生成的NC代码保存为可加工的txt文件。

## 使用软件及语言

### Qt简介

Qt是一个跨平台的应用程序和用户界面框架，广泛用于开发图形用户界面程序（GUI）。它由Qt Company（之前称为Trolltech）开发，并在LGPL或商业许可下发布。Qt支持多种编程语言，最常用的是C++，但也支持Python（通过PyQt）、Ruby等。

**主要特性**包括：

1. **跨平台**：Qt应用程序可以在多种操作系统上运行，包括Windows、macOS、Linux、iOS、Android等。
2. **丰富的API**：提供广泛的库和工具，用于创建复杂的GUI和访问系统功能。
3. **信号和槽**：Qt采用独特的信号和槽机制，简化了对象间通信。
4. **模型/视图编程**：提供了一种灵活的方式来处理数据和显示数据。
5. **网络编程**：支持网络请求、数据传输等。
6. **多语言支持**：易于国际化和本地化，支持多种语言。
7. **QML和JavaScript**：支持QML（Qt Modeling Language）和JavaScript，用于快速开发动态和流畅的应用程序。
8. **工具链**：包括Qt Creator IDE、Qt Designer等工具，帮助开发者设计和构建应用程序。

### C++简介

C++是一种通用的、静态类型的、多范式编程语言，由Bjarne Stroustrup创建，作为C语言的扩展。C++被设计成直接而高效的，同时提供高级特性，如面向对象编程、泛型编程和函数式编程。

**主要特性**包括：

1. **面向对象**：支持类和对象，允许封装、继承和多态。
2. **性能**：编译为机器代码，执行效率高。
3. **泛型编程**：通过模板提供泛型编程支持，允许编写与数据类型无关的代码。
4. **函数式编程**：支持函数式编程特性，如Lambda表达式、高阶函数等。
5. **内存管理**：手动内存管理，提供指针和引用，允许精细控制资源。
6. **标准库**：包含广泛的标准库，提供对文件输入/输出、数据结构、算法等的支持。
7. **异常处理**：支持异常处理机制，用于处理程序运行时的错误。
8. **编译器和工具链**：有多个编译器选项，如GCC、Clang、MSVC等。
9. **社区和生态系统**：拥有庞大的开发者社区和成熟的生态系统，提供大量的库和框架。

C++被广泛应用于系统软件、游戏开发、嵌入式系统、实时控制、桌面软件、服务器和高性能计算等领域。由于其性能和灵活性，C++仍然是当今最流行的编程语言之一。

# 非圆曲线的逼近算法与算法设计

## 非圆曲线插补算法的原理与实现

### 非圆曲线的插补算法

根据给定进给速度和给定轮廓线形的要求，在轮廓的已知点之间，确定一些中间点的方法，这种方法称为插补方法或插补原理。面向非圆曲线的插补逼近一般分为直线逼近和圆弧逼近这两大类，其中直线逼近常见的有等间距法、等弦长法和等误差法，其优点在于方法简单，易实现，且计算机计算速度快，但缺点也很明显，那就是段数过多；圆弧逼近一般采用曲率圆法、三点圆弧法和相切圆法等方法，其优点在于圆弧逼近的段数一般相比直线逼近要少很多，但缺点也很明显，那就是方法复杂，难以实现，且计算机需要处理的数据较多，计算得到拟合圆弧段的速度相比直线拟合逼近方法慢得多，不管哪种方法，其关键和难点主要在于寻找节点。一个已知曲线的节点数主要取决于逼近线段的形状（直线段还是圆弧段），曲线方程的特性以及允许的逼近误差。将这三者利用数学关系求解，即可求得一系列的节点坐标，并按节点划分程序段。使用者可以根据不同的情况，选取适合的方法，以满足不同的需求。

### 等间距直线逼近插补

等间距直线逼近算法（Equidistant Straight Line Approximation）是一种用一系列等间距的直线段逼近曲线的方法。该算法常用于数控加工中的轨迹规划和插补过程中。等间距直线逼近算法相对简单，容易实现和理解。它不需要复杂的数学运算或曲线拟合算法，适用于快速的轨迹规划和插补计算。通过调整步长或等间距距离，可以控制逼近后的直线段的数量。较小的步长可以得到更精细的逼近结果，但会增加计算量，而较大的步长可以减少直线段的数量，但可能会导致逼近误差增大。等间距直线逼近算法是一种近似方法，逼近后的直线段与原始曲线之间存在一定的误差。因此，在一些对精度要求较高的应用中，可能需要考虑使用其他更精确的逼近算法或进行后续的误差补偿处理。

等间距直线逼近是最简单的一种逼近方法。如图 2-1所示，已知方程，根据给定的求出求出，将代入即可求得一系列。，即为每个线段的终点坐标，并以该坐标值编制直线程序段。

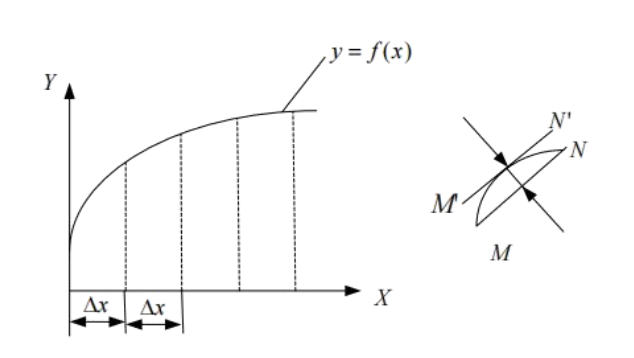


图 2-1直角坐标系下等间距直线逼近方法原理图

在极坐标上，我们也可以采用等间距法，对于给定的角度增量，取起点，往顺时针方向或逆时针方向，每隔距离，在曲线上取一点作为插补点，同时以参数方程的方式计算该点的坐标值，，最后再将相邻的线段拟合原曲线。

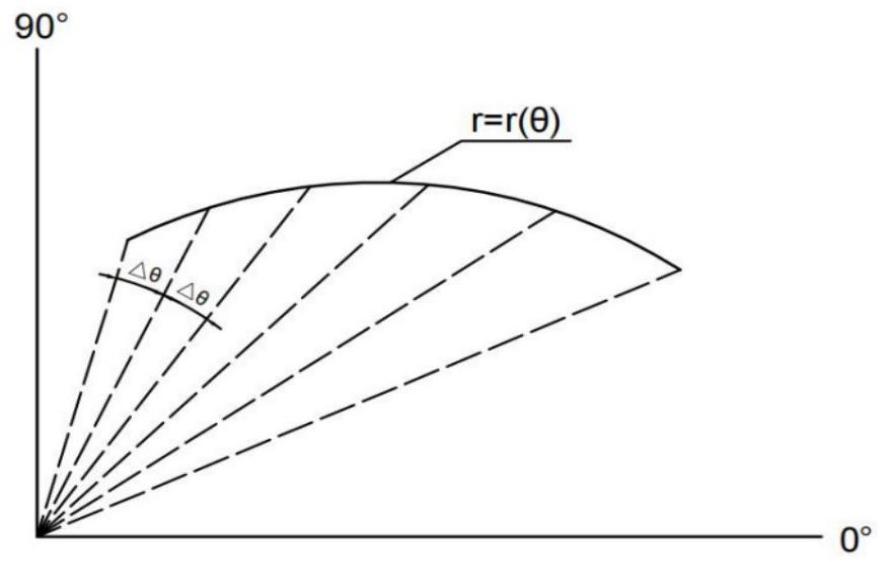


图 2-2极坐标系下等间距法原理图

取值的大小取决于曲线的曲率和允许误差。与两直线的距离为允许误差。的方程为，则的方程为。通过两方程联立得：

如果求解无解，即没有交点，表示逼近误差小于；如果只有一个解，即等间距与轮廓线相切，表示逼近误差等于；如果有两个或两个以上的解，表示逼近误差大于，这时应缩小等间距坐标的增量值，重新计算节点和验算逼近误差，直至最大的逼近误差小于等于。

误差校验上，我们通过对坐标的累加或累减，得到数个坐标，带入到曲线公式中，得到相应的坐标值，以此获得各插补点。等间距法计算简单，但由于取定值应保证曲线曲率最大处的逼近误差允许值，所以程序可能过多；插补段数可控，但对于曲率大的地方表现并不良好，误差不可控。等间距法计算简单，但由于取定值应保证曲线曲率最大处的逼近误差允许值，所以程序可能过多。用此种方法进行数学处理，它的逼近曲线与轮廓线的逼近误差参差不齐，程序明显增多，影响机床的加工效率，不适合大批量的加工，成本也比较高。

### 等弦长直线逼近插补

等弦长法的本质是使所有逼近线段的长度相等。如图 2-3，对于给定的弦长，我们在曲线上先取起点，在起点处做一以为半径的圆，该圆与曲线的交点作为下一个起点，继续做以为半径的圆，找下一个交点。以此类推，可以在曲线上得到一组插值点，且两两相邻的点所连成的直线，长度相等。由于圆与曲线有两个交点，需要往同一方向取点（顺时针或逆时针），避免重复取点。

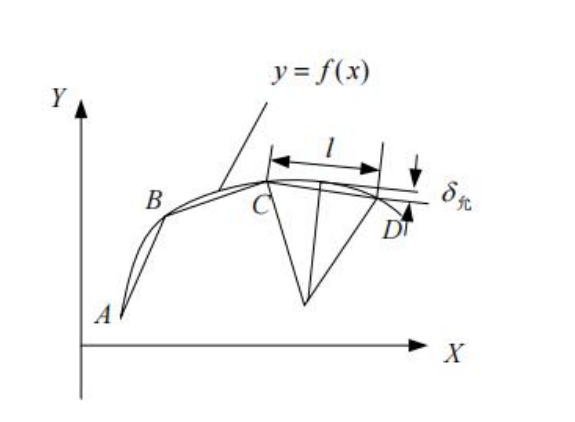


图 2-3等弦长直线逼近原理图

已知点，求得圆方程为，与曲线方程联立得：

即可求得下一个起点。

等弦长法对于曲线各处的曲率相差较大时，所求得的节点数过多，所以这种方法宜用于曲率变化不大的曲线节点计算。

### 等误差直线逼近插补

等误差直线逼近算法（Equal Error Line Approximation Algorithm）是一种用于将曲线逼近为直线段的方法，其特点是控制直线段的长度和方向，使每个直线段能满足预设的误差条件。

与等间距直线逼近算法相比，它们的主要区别在于逼近直线段之间的间距调整策略和逼近效果。等误差直线逼近算法通过根据每个控制点与逼近直线段之间的误差调整直线段的长度和方向，以实现更精确的逼近。这种方法的优点是能够根据曲线的变化情况进行动态调整，使得逼近误差相对均匀分布，更符合实际曲线的形状。然而，等误差直线逼近算法可能需要更复杂的计算和迭代过程，以得到逼近结果。

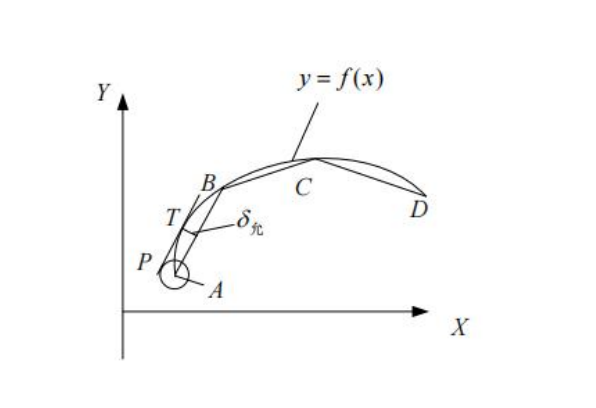


图 2-4等误差直线逼近原理图

等误差直线逼近算法的基本思想和步骤：

1. 确定所有逼近线段的误差的圆的方程，即以起点为圆心，为半径作圆方程：

将方程写为

。

1. 求与曲线的公切线的斜率：
2. 上式中的、、、通过下列联立方程求得：
3. 求弦长的方程。使弦的斜率为，即使平行，则方程为：
4. 联立曲线方程和弦方程求得点坐标：
5. 重复上述步骤即可顺次求得曲线上各个节点的坐标。

对于曲率变化较大的曲线，用等误差法求得的节点数最少，但计算稍繁。

## 幂函数（指数）曲线的逼近算法

### 等间距直线逼近算法

对于一般的幂函数，其方程为：

，

其中、、为参数，为幂函数的指数，其值介于0至1之间，是此幂函数的定义域左端点值（默认状态下为1），是此由用户进行输入。对其采用等间距直线插补算法进行直线逼近，如图 2-5。

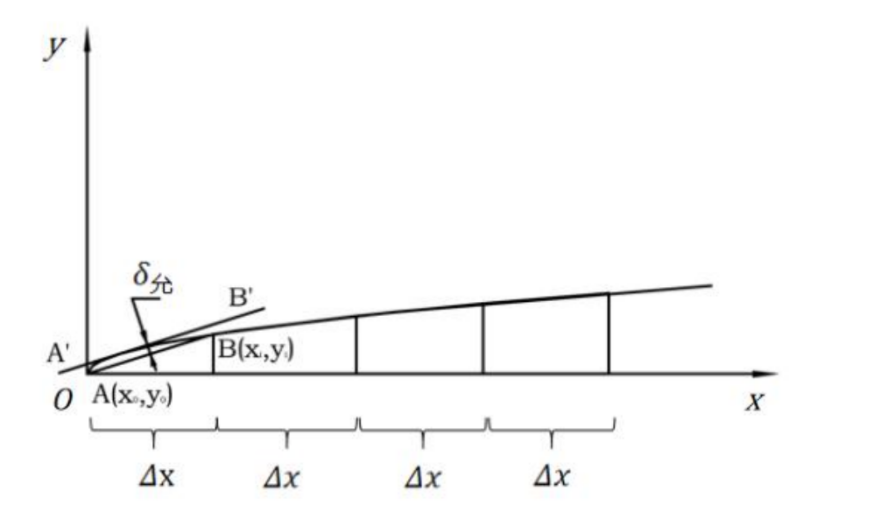


图 2-5幂函数曲线等间距直线插补算法原理图

已知起点，给定的步长，幂函数曲线的曲率和允许误差,求出下一个节点横坐标，并求出下一个节点纵坐标。

从而得下一个节点坐标，由、两点可得线段的斜率:

直线的方程为：

整理后的方程为:

通过联立上述公式求得的斜率，可以得到与直线斜率相同且与幂函数相切的直线与幂函数相切的坐标，即：

点到直线的距离公式为：

将方程进行整理，利用点到直线的距离方程以及切点的坐标，即可求出切点到直线的距离，即为逼近误差。

重复上述步骤，以上一段直线段的终点作为下一段直线段的起点，依次求出各个拟合直线段的逼近误差，通过冒泡排序法即可求出直线段拟合非圆曲线中的最大逼近误差。其算法流程图如图 2-6所示。

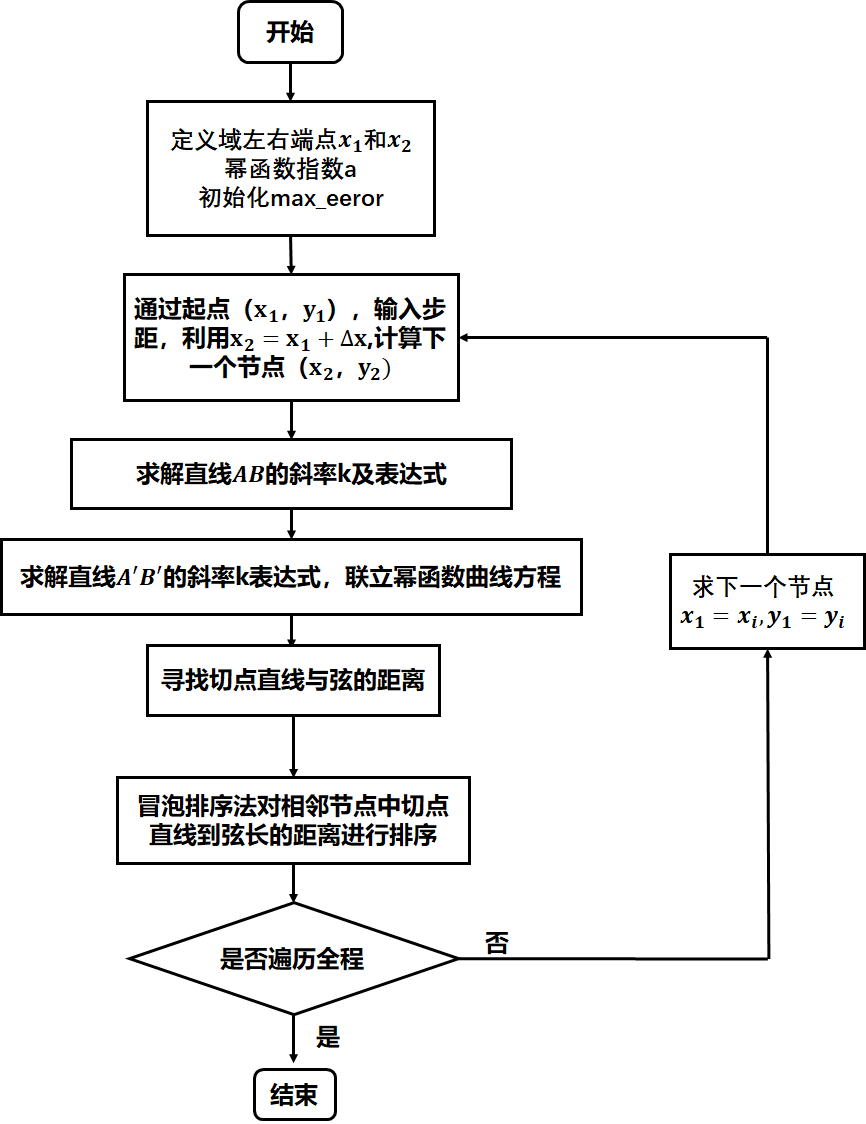


图 2-6 等间距插补算法流程图

优化思路

由于取值的大小取决于曲线的曲率和允许误差。设定一个允许误差，当其最大逼近误差大于允许误差时，减小步距。与两直线的距离为允许误差。通过两方程联立得：

如果求解无解，即没有交点，表示逼近误差小于；如果只有一个解，即等间距与轮廓线相切，表示逼近误差等于；如果有两个或两个以上的解，表示逼近误差大于，这时应缩小等间距坐标的增量值，使用for循环重新计算节点和验算逼近误差，直至最大的逼近误差小于等于。其算法流程图如图 2-7所示。

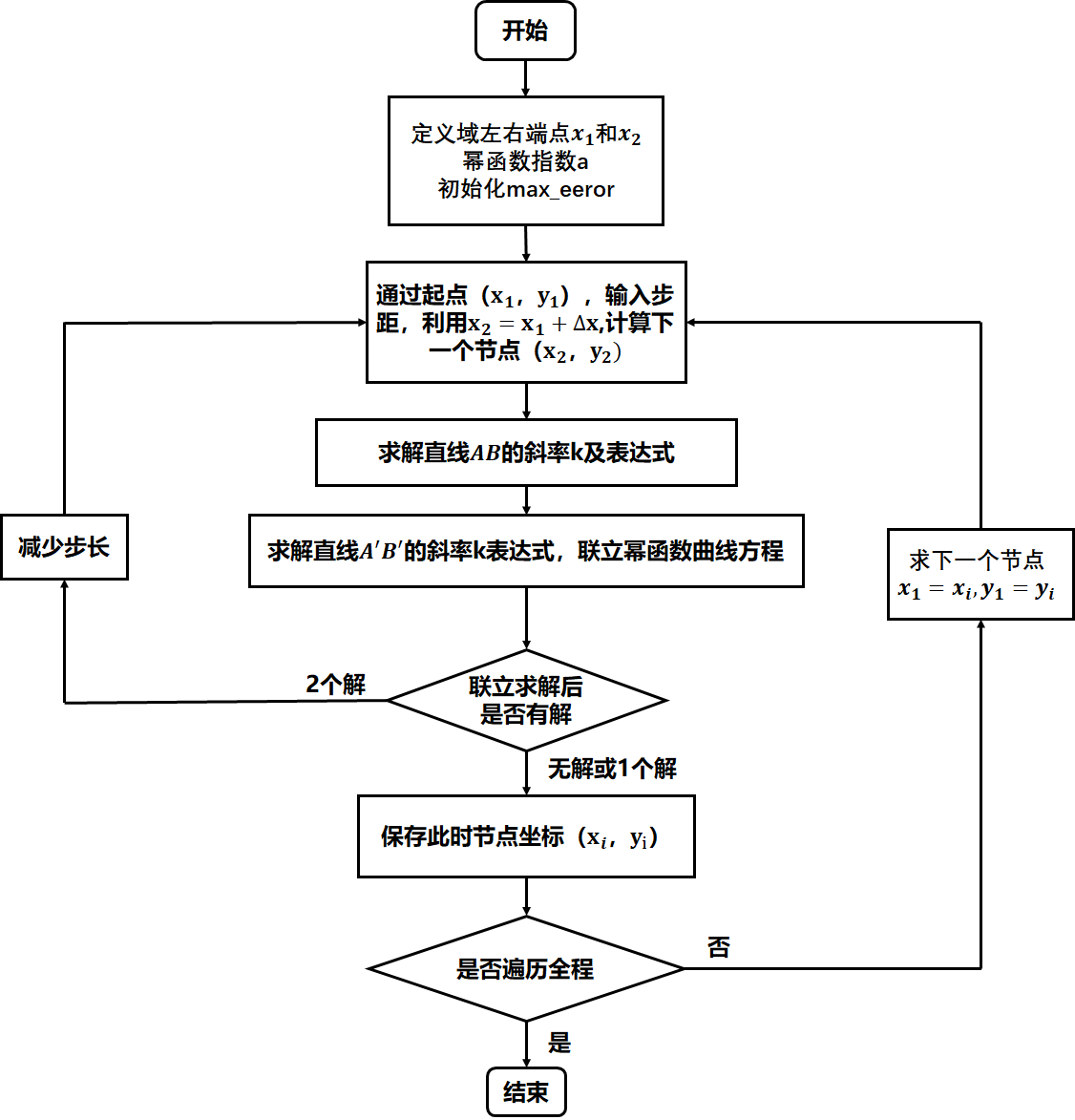


图 2-7幂函数等间距插补算法流程示意图

### 等误差直线逼近算法

对于一般的幂函数，其方程为：

其中、、为参数，为幂函数的指数，其值介于0至1之间，是此幂函数的定义域左端点值（默认状态下为1），是此由用户进行输入。对其采用等误差直线插补算法进行直线逼近（图 2-8）。

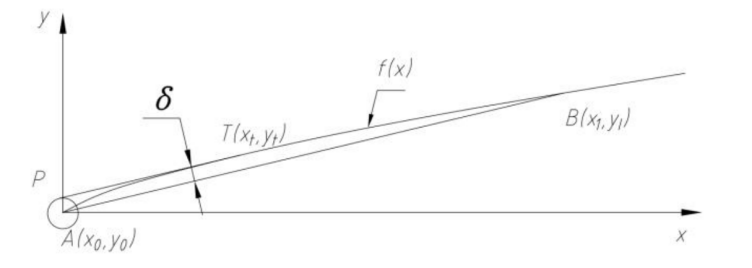


图 2-8幂函数曲线等误差直线插补算法原理图

确定所有逼近线段的误差的圆的方程，即以起点A为圆心，为半径作圆方程：

求与曲线的公切线的斜率：

上式中的、、、通过下列联立方程求得：

式中—误差圆函数，—加工曲线函数

求弦长的方程。使弦的斜率为，即使平行，则方程为：

联立曲线方程和弦方程求得点坐标：

重复上述步骤即可顺次求得幂函数上各个节点的坐标。

上述联立方程求解的实质在于通过已知幂函数曲线上的节点来确定下一个节点的位置，这一过程中涉及到计算已知节点到幂函数下一节点处切线的距离。我们的目标是判断这个距离是否满足预定的允许逼近误差要求。具体来说，我们首先需要确定已知节点到切线的距离，然后比较这个距离与允许的逼近误差。

如果计算出的距离小于或等于允许逼近误差，那么我们可以认为当前的节点已经足够接近真实的解，无需进一步调整。然而，如果计算出的距离大于允许逼近误差，那么我们就需要对下一个节点的位置进行调整。

在等间距的初始设定下，如果计算得到的逼近误差大于允许逼近误差，我们不会直接接受这个节点作为解。相反，我们会返回到上一个节点，并将步距（step）减半，然后再次尝试计算下一个节点的位置。通过这种逐次迭代的方法，我们可以逐步缩小逼近误差，直到算得的误差与允许逼近误差的差值小于一个很小的阈值（如0.0001）。当这个条件满足时，我们可以认为当前节点已经足够接近真实的解，并将其作为近似解接受。

优化思路：

在利用等误差算法来求解幂函数的过程中，步距的大小对迭代速度和求解精度具有直接影响。如果步距设置得过大，虽然能够加快收敛速度，但可能会牺牲求解的精度；反之，如果步距设置得过小，虽然能够确保求解的精度，但会减慢收敛速度。因此，为了同时兼顾收敛速度和求解精度，我们需要动态地调整步距。

在迭代过程中，每次增加步距后，我们会检查当前解的精度是否达到要求。如果未能达到预定的精度要求，我们会将步距加倍，以增加收敛速度，同时尝试更快地接近真实的解。然而，如果当前解超出了精度要求，即逼近误差过大，我们会将步距减半，进行更细致的逼近，以保证求解的精度。

通过这种动态的步距调整机制，我们可以逐步缩小逼近误差，直到算得的逼近误差与允许逼近误差的差值小于0.0001。此时，我们可以认为当前节点已经非常接近真实解，并将其作为近似解接受。其算法流程图如图 2-9所示。

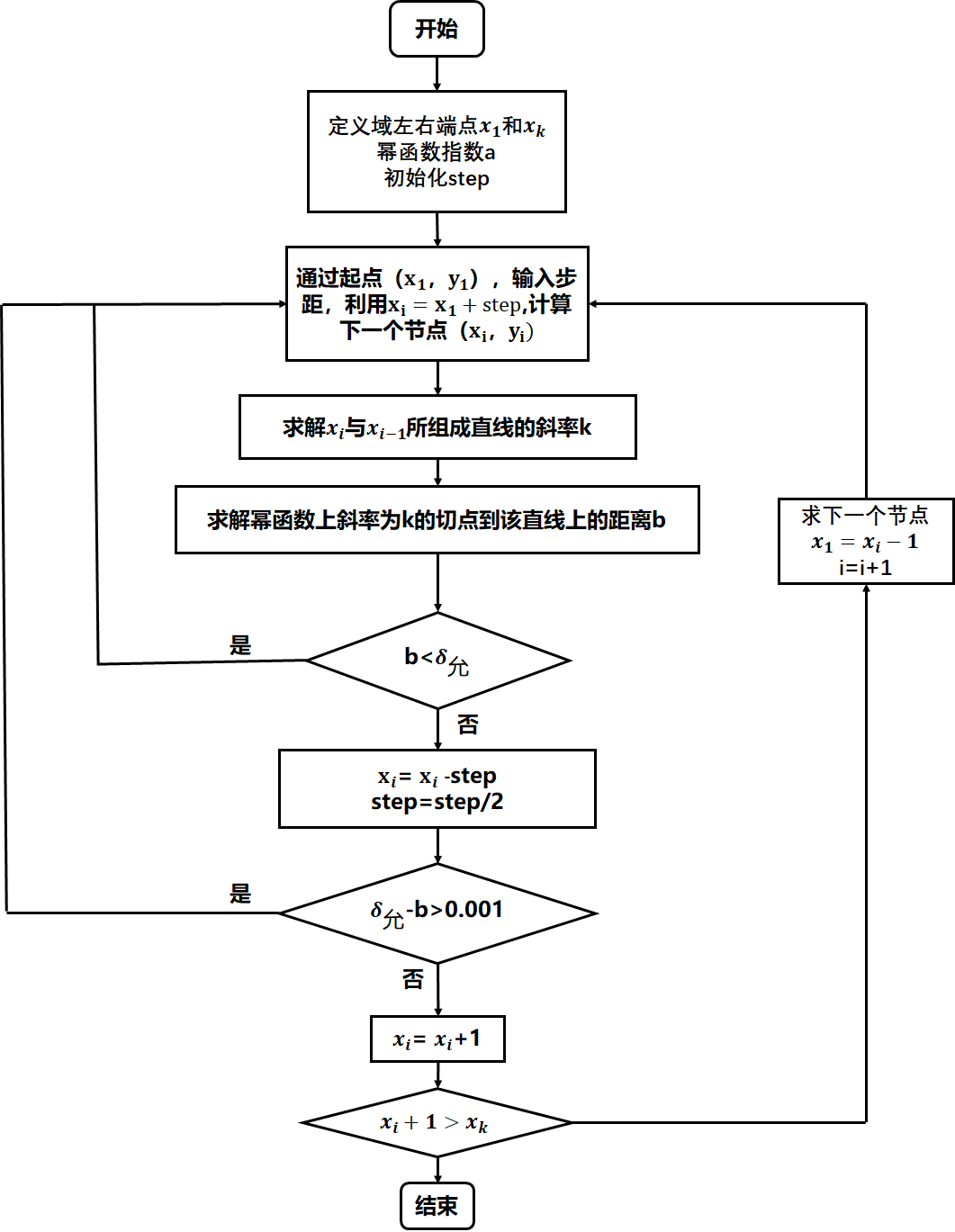


图 2-9等误差直线逼近算法流程图

## 余弦加减速盘形凸轮轮廓的逼近算法

通过机械原理“9-2凸轮的运动规律”可知，凸轮余弦加减速推程段的运动方程为：

凸轮余弦加减速回程段的运动方程为：

本文研究的是的对心直动尖顶推杆盘形凸轮机构，在考虑近休止和远休止阶段，将凸轮运动分为四个阶段，即为推程、远休止、回程和近休止，记基圆半径为，升程为，凸轮升程角为，远休止角为，回程角为，近休止角为，凸轮每个阶段开始的回转角为，用直角坐标方程表示各阶段的轮廓线方程。

升程：

远休止：

回程：

近休止：

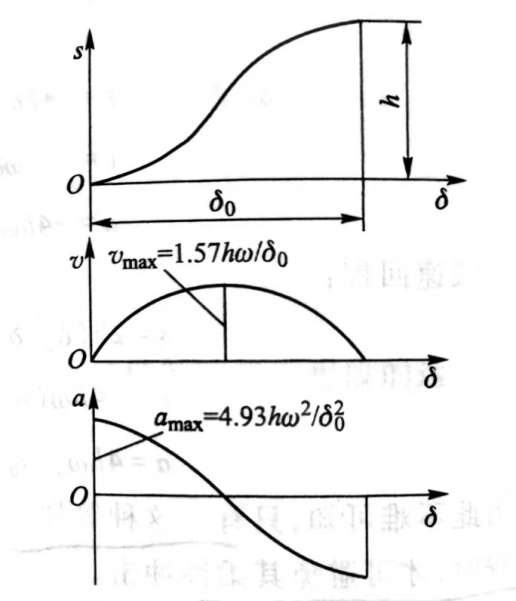


图 2-10余弦加减速盘形凸轮推程运动规律

### 等间距直线逼近算法

对于给定的起点即和确定的步长（或）,计算出下一插值点的角度（或横坐标），得到下插值点坐标，对前后两插值点进行连线。相关计算如上文介绍四个阶段轮廓线和坐标方程。

由理论轮廓线方程式直角方程式知，对给定基圆半径为，升程为，凸轮升程角为，远休止角为，回程角为，近休止角为，可以得到理论轮廓方程式，在进行等间距直线插补时运用的是步进角。根据凸轮的余弦加减速运动规律，本项目将其分为四个阶段，即推程、远休止、回程和近休止，，如前所述，可以得到每个阶段凸轮的理论轮廓方程式，每个阶段的终点即为下个阶段的起点。对于每个阶段，我们采用了等间距直线逼近方法进行处理，具体实现与幂函数的等间距直线逼近方法类似，这里就不再详述。单个阶段的算法流程图如下图所示。

优化思路

在计算与匀加减速曲线相切的直线与相邻两个节点所在的弦之间的距离时，我们采用一种冒泡排序迭代方法。通过比较每次计算出的误差值与我们设定的允许误差值，我们可以判断当前的步长是否足够精确。如果计算出的误差值大于允许误差值，我们会减小步长并重新进行迭代计算，直到误差值降低到允许误差值以下。这样的过程确保了计算的精确性，并确保了最终的弦与直线之间的距离满足我们的精度要求。其算法流程图如图 2-11所示。

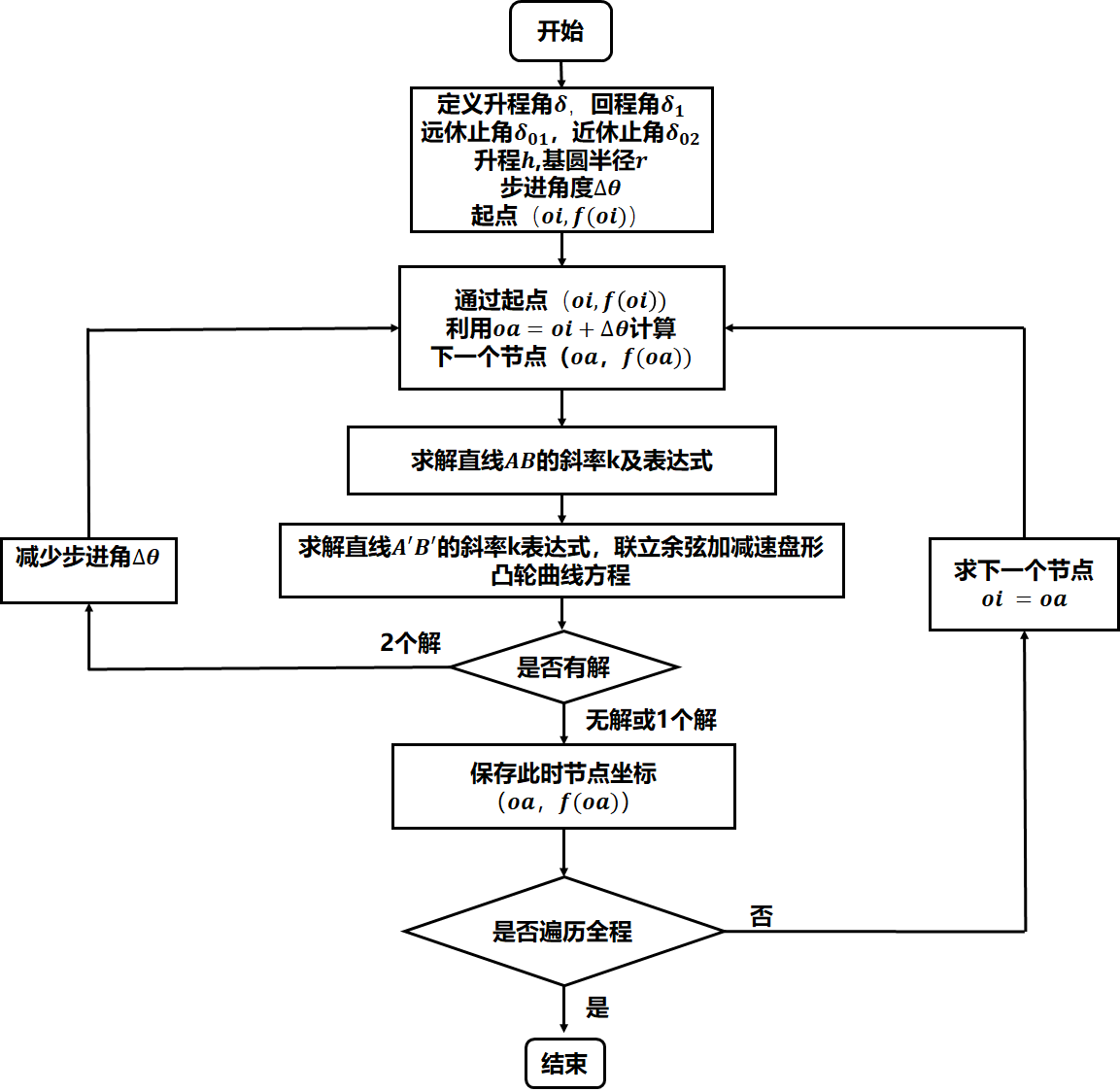


图 2-11盘形凸轮等间距逼近算法流程图

### 等误差直线逼近算法

等误差的本质是通过误差控制的方法，使得所有插补线段的最大误差相等。在2.1.4节已经具体阐述了等误差直线插补逼近，对于凸轮曲线，其基本原理类似，不同在于本文研究的余弦加减速凸轮运动分为四个阶段，且每一个阶段的和的直角坐标方程不一样。因此，在进行依次插补各个阶段的轮廓时，对每个阶段的轮廓做好首尾相连的处理。本文是采用采用步进角的方式进行步进。并且在求插补直线与凸轮曲线的交点时，采用二分法来对节点快速求解得到下一个插补点。

综上，如图 2-12为余弦加减速盘形凸轮轮廓线的等误差逼近算法流程图，其中步骤“求插补节点C”即利用到二分法求交点，上述已阐明具体过程，在流程图中不在赘述。

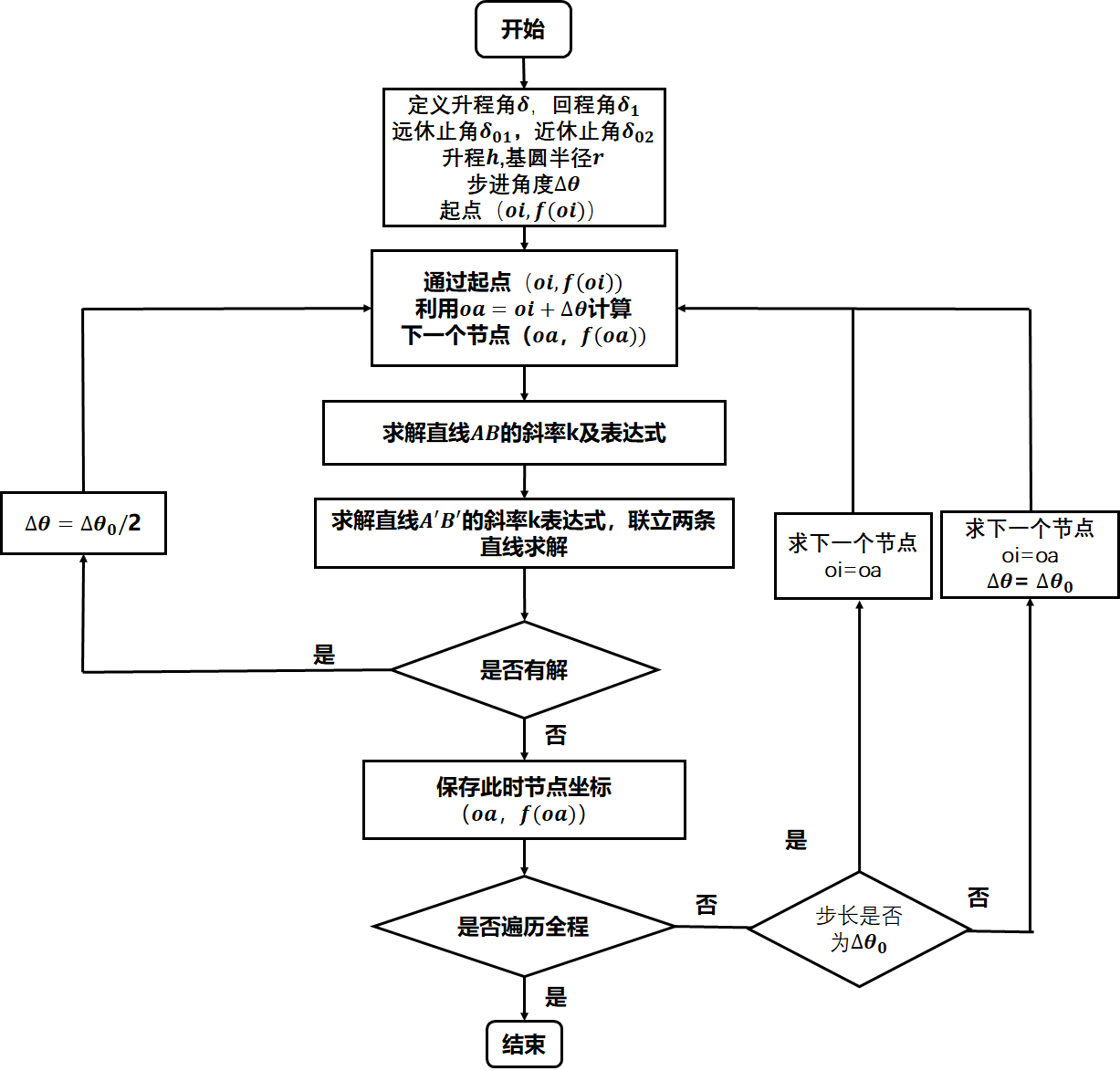


图 2-12盘形凸轮等误差逼近算法流程图

# 实现的功能

## 刀补功能

当前，我们在加工过程中普遍采用的刀具半径补偿主要有B型刀补和C型刀补两种。

B刀补在刀具中心轨迹的段间连接上采用了圆弧过渡的方式，这种算法简单易于执行。然而，由于它使用了圆弧作为段间过渡，这就带来了一些无法忽视的不足，比如处理尖角时工艺性较差，编程过程相对繁琐，并且容易因为刀具干涉而导致工件损坏。

C刀补采用了直线来连接相邻两段轮廓的刀具中心轨迹。数控系统会基于零件的编程轨迹和刀具偏移量直接计算出刀具中心轨迹的转接点，然后对原编程轨迹进行必要的伸长、插入或缩短修正，从而实现两个程序段间刀具中心轨迹的平滑转接。C刀补的主要优势在于它使用直线作为轮廓间的过渡，因此在处理尖角时，其工艺性相比B刀补更优越。现代CNC系统几乎都倾向于采用C刀补，这样编程人员可以完全按照零件的实际轮廓进行编程。

在本项目中，我们优先选择C刀补作为刀具补偿方法，该方法基于矢量算法（特指非加工侧的夹角）进行计算。根据任务书的要求，我们需要在直线插补和圆弧插补中选择一种，本项目决定采用直线插补的直线逼近方法。这种刀补方式的特点是直线-直线的建立方式，而转接方式则根据矢量夹角的不同，分为三种类型：当时为伸长型转接，当时为缩短型转接，以及当时为插入型转接。为了更具体地描述这一过程，我们定义加工侧的夹角为转接角，并基于转接角的不同，在图 3-1中展示了五种类型的转接点算法，以便详细讨论刀补建立过程中的转接点计算方法。

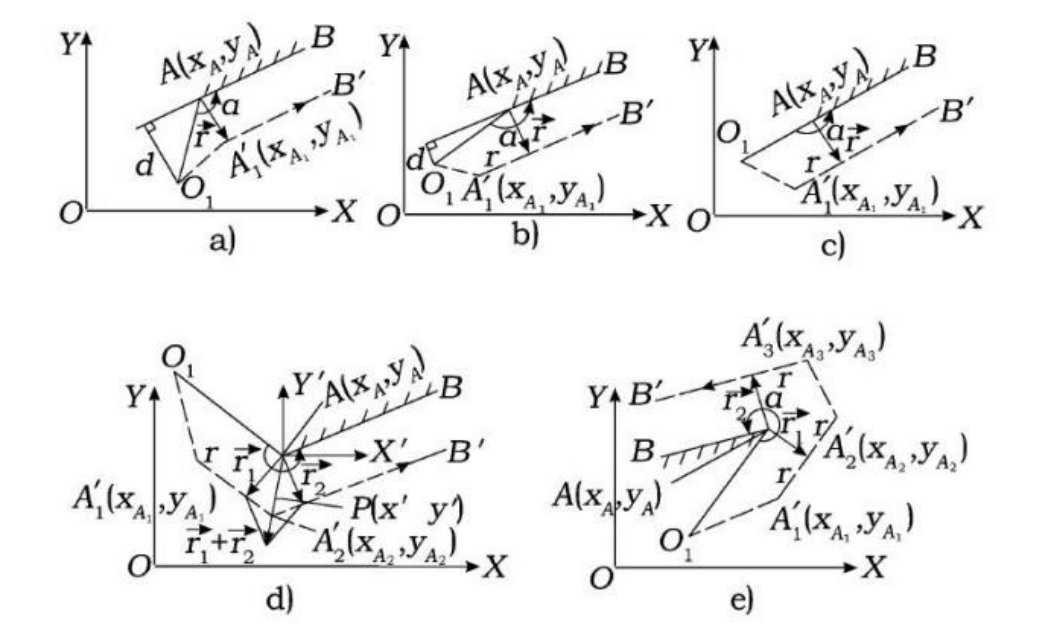


图 3-1直线-直线的刀补转换类型

图3-1直线-直线的刀补转换类型

（1）矢量夹角时

有两种转接情况，其一是如图 3-1a所示（起点+中间过渡段），当刀补建立线段起点到直线的距离(为刀具半径,为避免开方运算,可用代替)时,此时为缩短型；其二如图 3-1b所示，当时,此时为伸长型。

当为缩短型时,转接点坐标为：

当为缩短型时,转接点坐标为：

式中为垂直于线段的刀具半径矢量的分量。

（2）矢量夹角时

当时,为伸长型。其中图 3-1c为矢量夹角为的情况。通过伸长型的公式，可求得其转接点的坐标。图 3-1d为时的伸长情形,转接点的坐标计算公式如下:

式中为垂直于刀补建立段的刀具半径矢量的分量，转接点为为：

其中为上到距离为的点的坐标。

（3）矢量夹角时

当时，如图 3-1e所示，为插入型。三个转接点的坐标为：  
，

，

，

刀补在加工流程中分为三个阶段：首先是刀补建立，它设定了刀补的基本参数和矢量路径；接着是刀补进行阶段，这一过程中，系统会读取前两次加工段的信息，并依据已确定的矢量路径，在每次加工段的前行方向左侧或右侧添加相应的刀补矢量，以实现精确的加工；最后是刀补撤销阶段，这一阶段实质上是对刀补建立的逆向操作，旨在撤销先前所做的刀补调整，使加工状态恢复到未进行刀补调整前的状态。

## 防过切功能

如图 3-2所示，在加工特定曲线时，若参数设置不当或刀具半径选择过大，转接角处可能发生过切现象，这对加工质量构成潜在威胁。为避免这种情况，我们设计了一个过切自动报警功能。该功能基于一个关键观察：当转接角发生过切时，刀具中心轨迹会出现交叉，这会导致存储插补节点横坐标的数组中产生“回头”现象，即数组内会记录刀具中心重复走过的已插补区域的数据。为了及时检测并预警此类过切情况，我们采用了遍历数组的方法，通过分析数据模式来预先判断是否存在过切风险，从而触发自动报警机制。

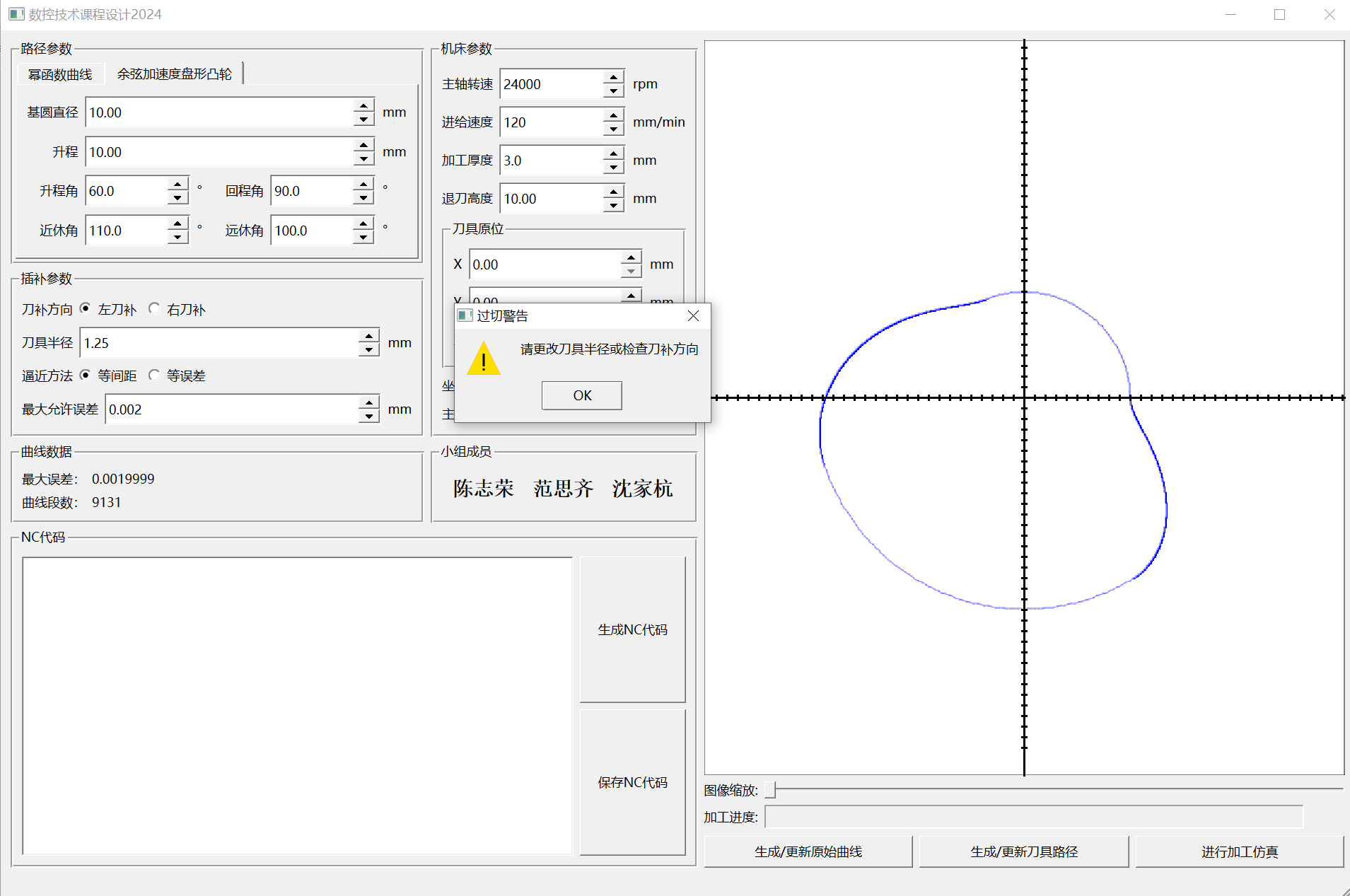


图 3-2过切报警功能

## NC代码生成和保存功能

NC代码生成和保存功能是数控编程中至关重要的环节。在数控加工过程中，CAD/CAM软件首先根据零件的设计图纸和加工工艺要求，自动生成NC代码，即指导数控机床进行加工操作的指令集。如

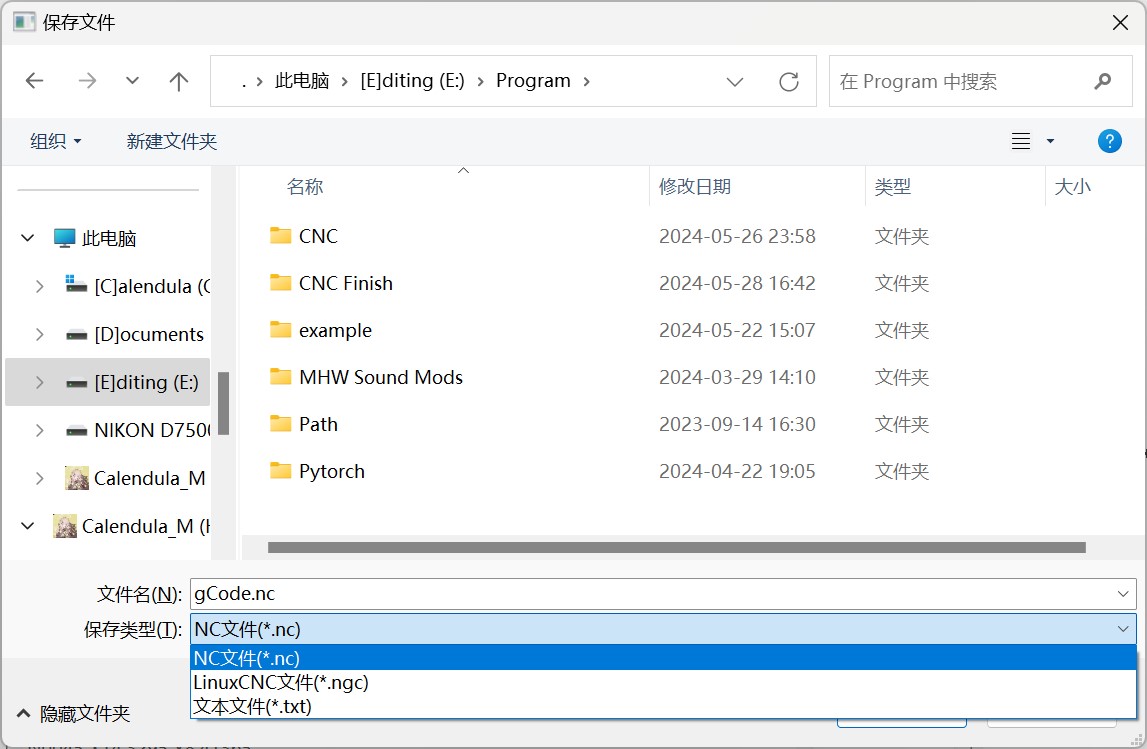


图 3-4代码保存功能

这些代码包含了切削速度、进给量、刀具路径等关键信息，确保加工过程的精确性和高效性。

一旦NC代码生成完毕，系统便提供了保存功能，如图 3-4和图 3-5，允许用户将代码保存为特定格式的文件，如.txt或.nc文件，以便后续使用和传输。保存过程中，系统会对代码进行校验和整理，确保代码的正确性和可读性。

NC代码的生成和保存功能不仅提高了编程效率，还确保了加工过程的一致性和可追溯性。用户可以随时调用已保存的NC代码，进行修改或再次使用，极大地方便了数控加工的生产和管理。

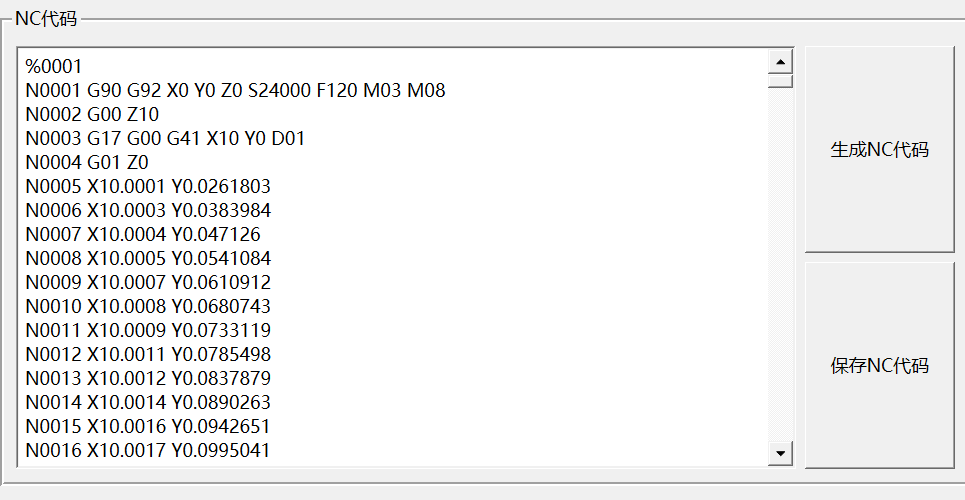


图 3-3NC代码生成功能

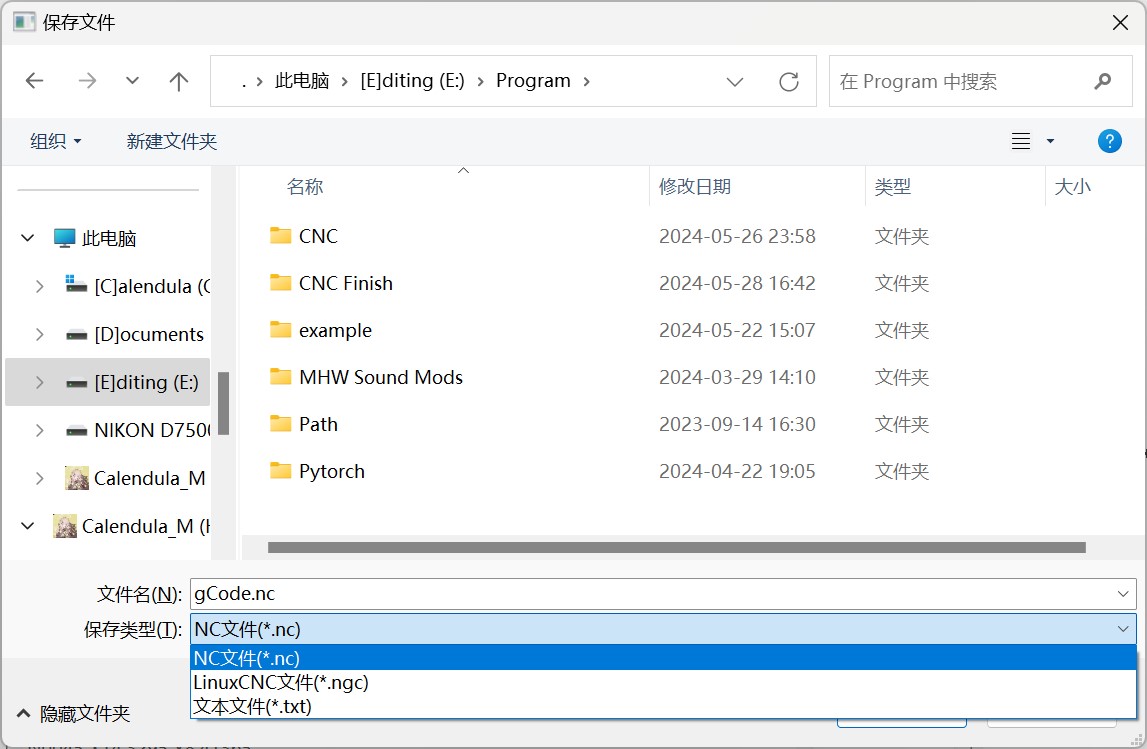


图 3-4代码保存功能

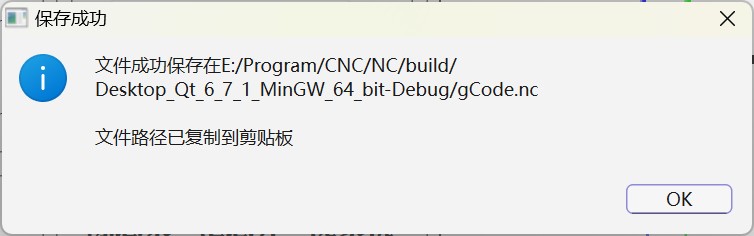


图 3-5NC代码保存成功

## 模拟加工功能

根据先前确定的加工过程中刀具中心的运行轨迹，我们可以进一步模拟刀具的实际运动情况。在已知刀具半径的条件下，我们选取刀具中心轨迹上的点作为圆心，以刀具半径为圆的半径，绘制出一系列红色的圆。这些圆所组成的图案能够直观地模拟出刀具在加工过程中的运行轨迹，并且我们还弄了进度条，能够直观的看见加工进度。具体效果如图 3-6所示。

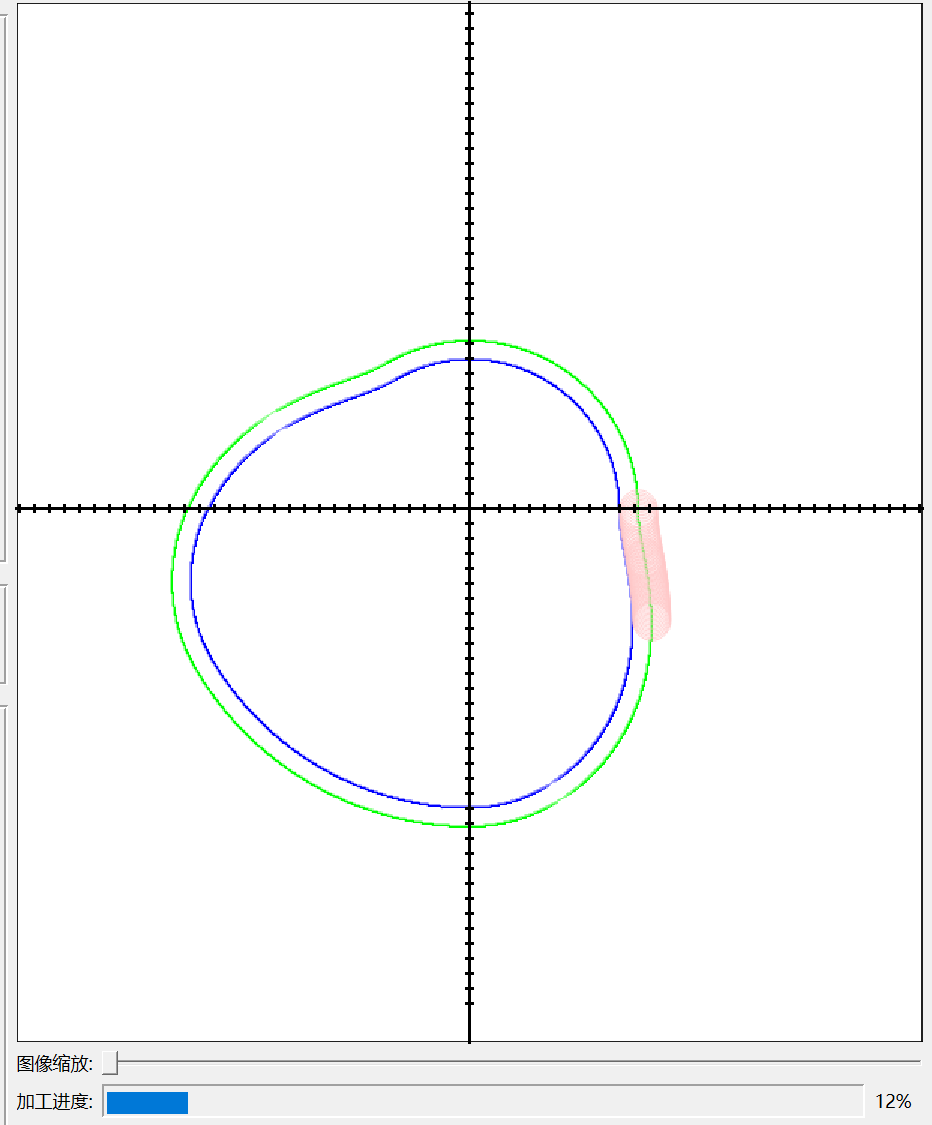


图 3-6模拟加工功能

## 绘图区域缩放

# 软件界面

## 主界面总体介绍

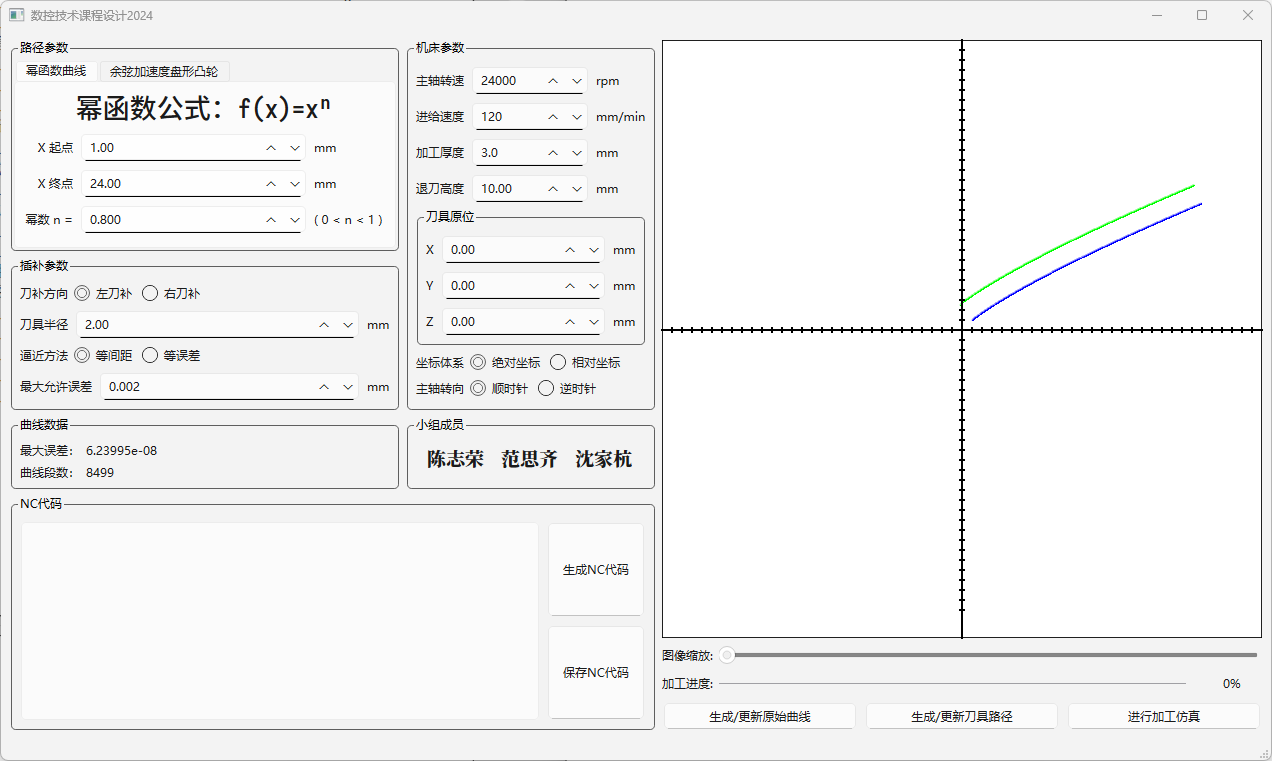


图 4-1主界面

软件主界面如图 4-1所示，包括路径参数设置区，插补参数设置区，曲线数据显示区，机床参数设置区，小组成员区，NC代码操作区，绘图区，主操作按钮区和加工进度条。在软件启动时所有数据输入框都被赋值为一个较为合理的值，便于在用户不理解参数时也可以先生成图像，在改变参数并观察新图像后理解该数据的作用，而不是对着满屏幕的空白输入框尝试输入一系列可能非法的数据。

软件的基本操作顺序为：选择曲线路径→设置曲线参数→设置插补参数→点击“生成/更新原始曲线”按钮检查原曲线→点击“生成/更新刀具路径”按钮检查刀具路径→点击“进行加工仿真”检查仿真过程→设置机床参数→点击“生成NC代码”按钮检查NC代码→点击“保存NC代码”按钮以想要的格式保存代码至想保存的地方→根据剪贴板中的文件路径打开NC文件。

其中拖动绘图区下方的“图像缩放”按钮会以坐标原点为中心缩放整个绘图区域，主操作按钮区域上方的”加工进度“进度条会在进行加工仿真时显示当前仿真的进度。

下面是每个功能区的详细介绍：

## 路径参数设置区

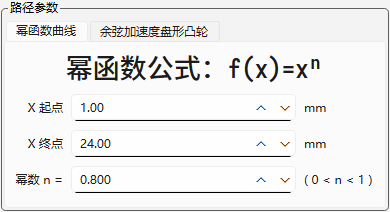


图 4-2幂函数参数

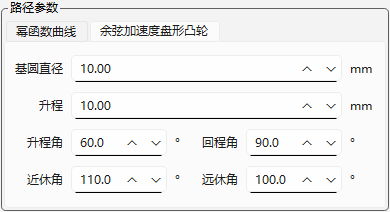


图 4-3凸轮参数

如图 4-2与图 4-3所示是路径参数设置区域，在该区域内可以输入幂函数曲线或余弦加速盘形凸轮轮廓的基本参数。对于幂函数曲线，参数为X的起点终点及幂函数的幂数，我们限定了X的起点比终点的值小且幂数在0到1的范围内，无法输入其他值。对于余弦加速盘形凸轮，参数为基圆直径、升程、升程角、回程角、近休角和远休角，且当四个角度之和不为360°时，会弹出错误提示框。

### 限定起终点大小

图4-4 限制起终点大小

图 4-4限制起终点大小

由于算法原因，我们只能计算X单调递增的正X值幂函数。如图 4-4，我们通过Qt中doubleSpinBox的setRange函数限制了X的值只为正数且小于200。因为对于大多数屏幕，这个窗口应该无法显示超过200的X值，原因将在UI设计补充4.11中叙述。

而对于单调递增的限制逻辑，我们将在4.9.1中叙述。

### 限定幂数大小

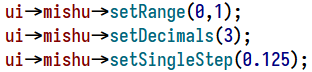


图 4-5限制幂数大小

我们组的题目为指数为1/4或1/8的幂函数，即指数为0.25和0.125。所以我们利用同样方法设置了幂数只能在0到1之间，如图 4-5所示，为了用户方便操作，我们利用setDecimals函数为这个doubleSpinBox设置了一个三位小数的精度，且使用setSingleStep函数保证用户在不想亲自输入数据时只需点击箭头就可以很方便地以固定的步长改动数据，每点一次箭头，该值会变动0.125。

### 限制角度



图 4-6限制角度

我们组的题目为余弦加速度盘形凸轮，如图 4-6所示，我们初步限制了四个角度输入框的取值范围，避免输入非法的参数，为了方便用户操作，我们使用了同样的方法设置了视觉上较为合适的精度和点击箭头改变数值的操作。

此外，如果四个角度之和不等于360°，则这组数据也是非法数据，这一判断的逻辑和操作将在4.9.1中叙述。

## 插补参数设置区



图 4-7插补参数区域

如图 4-7所示是插补参数设置区，在该区域内可以输入刀具路径的各种参数。可以选择使用左刀补或右刀补，并输入刀具半径；可以选择等间距逼近或等误差逼近，对于等间距逼近，可以设置最大允许误差，而对于等误差逼近，可以设置这个误差，如图 4-8所示。



图 4-8等误差设置

在选择等间距逼近时，误差设置框自动隐藏，选择等误差逼近时，最大允许误差设置框自动隐藏，避免用户额外输入不需要的参数或输错参数。

### 刀具半径输入优化

图4-10 刀具半径输入

图 4-9刀具半径输入

如图 4-9，由于常见的刀具半径多为0.25的倍数（虽然φ1.2和φ1.6的刀具也是常用刀具，但这个半径也支持手动输入，无伤大雅），我们用与之前一样的方法设计了刀具半径输入框的交互逻辑，方便用户操作。

### 逼近方法及其参数设置逻辑

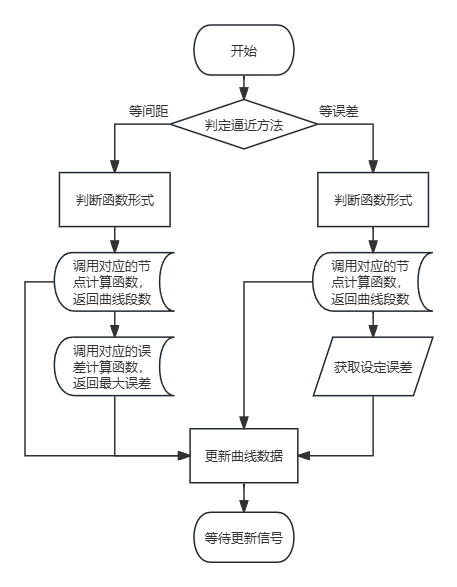


图 4-10逼近方法更新

如图 4-10所示，由于几个计算函数的返回值是曲线段数或最大误差，正好可以更新曲线数据显示区中的数据，于是一开始我收到后端同学的代码时，打算直接在选择逼近方法时就调用计算函数，便写了这段逻辑。



图 4-11逼近方法选择窗口

由于这段逻辑包含了计算函数，是最终曲线生成的先决条件，当时为了保证用户进行了这个操作，设置了在用户改变路径类型时自动取消选择逼近方法的逻辑。由于Qt中radioButton的逻辑，这里需要一个不可见的radioButton来进行选择的取消操作。如图 4-11所示，有一个默认选中的radioButton。它被设置成不可见的，且其按下动作连接到了选择曲线窗口的窗口变化动作，如图 4-12第一行所示。

在报告定稿前这段逻辑已经显得略为冗余，它的存在仅仅只是为了让用户在更换逼近方法时能马上在曲线数据显示区看到作出更改后路径的各种参数变化，但因为我在开发时将这份代码保存在GitHub上并实时更新，为了避免有人不去理解直接照搬我这份代码，也为了让认真研究了我代码的同学们在紧张的课设中稍稍放松一下心情，我还是决定将这个还蛮可爱的小彩蛋保留在这里。

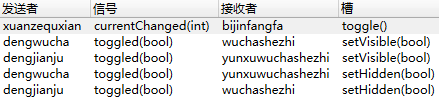


图 4-12逼近窗口信号

如图 4-12下面四行所示，当等间距被选中时，最大允许误差窗口显示，误差设置窗口隐藏；当等误差被选中时，最大允许误差窗口隐藏，误差设置窗口显示，避免用户在错误的地方输入数据。

## 曲线数据显示区

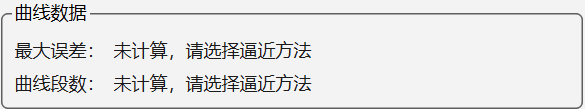


图 4-13曲线数据显示

该区域用于显示由当前选中和设定的曲线数据计算得出的曲线最大误差和曲线段数，主要逻辑已在4.3.2中叙述，此处便不多赘述。

值得一提的是，该区域的函数逻辑同样起到了调用计算函数的作用，使得在绘图函数中不需要计算曲线路径，直接取出计算好的路径并绘制即可。但也导致了在改动了其它参数但未改变逼近方法时点击“生成/更新原始曲线”按钮，曲线数据不会及时更新，需要更改逼近方法并改回才会更新数据。但这一逻辑也非常容易修改，直接把整条逻辑使用if函数写在“生成/更新原始曲线”按钮的点击事件中即可。

在报告定稿之前这一条逻辑已经被我加上去了，但我认为这条在撰写报告中产生的思考非常值得留在报告中，故没有将这段文字删除。

## 机床参数设置区

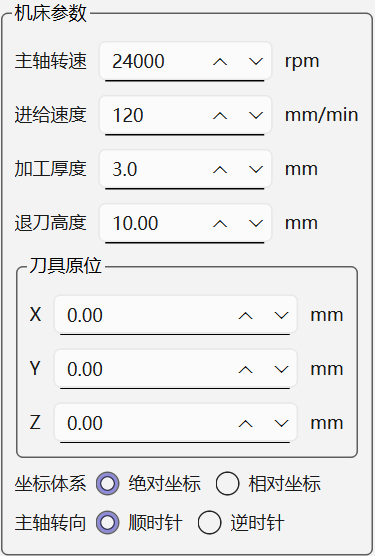


图 4-14机床参数区

图 4-14是机床参数设置区域，用户可以在这里设置机床的主轴转速、进给速度、加工厚度、退刀高度、刀具在开始加工前所在的位置、坐标体系和主轴转向。这个区域的设置除了“进给速度”之外只影响NC代码的生成，对原曲线、刀具路径和加工仿真的绘制不会有影响。

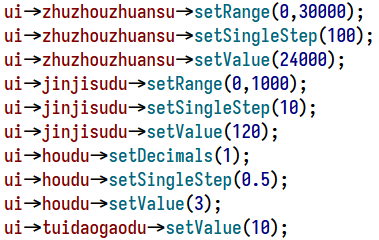


图 4-15机床参数初始化

如图 4-15，我们用了与之前同样的方法优化了该区域的用户操作。主轴转速的变更步长为100，进给速度为10，这个设置比较符合现实情况。此外，这两个参数一般只会是整数，所以我们使用了采样并输出int类型的spinBox而不是其他的doubleSpinBox来一定程度上回收该软件的内存占用。

值得注意的是，在具体的加工过程中，退刀高度应该高于毛坯厚度，也就高于加工厚度，在软件里我也对这个写了判断逻辑，这条逻辑将在4.7.1中叙述。

## 小组成员区



图 4-16小组成员窗口

讲了这么多烦人的代码和交互逻辑，让我们休息一下。如图 4-16，就像这个区域的名字所述，这只是一个用于按照学号顺序显示组员名字的区域，代表这玩意是我们做的。

## NC代码操作区

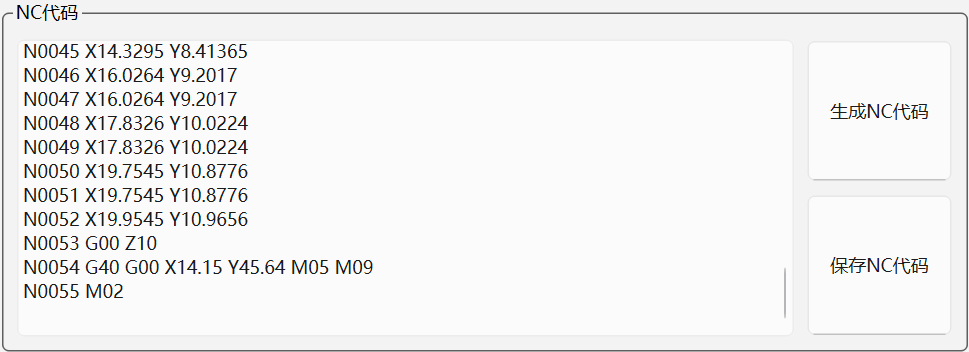


图 4-17 NC窗口

如图 4-17，这个区域有一个用于显示和微调NC代码的滚动文本窗，一个用于生成NC代码的按钮和一个用于保存检查并修改过的NC代码的按钮。在点击“生成NC代码”按钮之后，程序会在文本窗中输出NC代码，用户可以在这里检查并修改NC代码，在修改完之后点击“保存NC代码”按钮即可保存当前文本窗上的所有内容为一个文本格式的文件。在保存成功后程序还会自动复制文件的保存路径以便用户可以在保存后快速找到自己保存的NC代码文件。

### NC代码生成

#### 检查参数

在之前的图 4-14中我们能看出来，机床设置中有非常多参数需要设置，在使用中难免会有非法数据的输入，所以我们就需要在生成NC代码之前先检查可能的非法数据输入，比如退刀高度小于工件厚度或者未输入各种参数等情况。

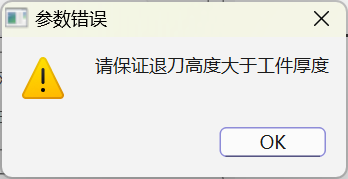


图 4-18退刀高度非法报警

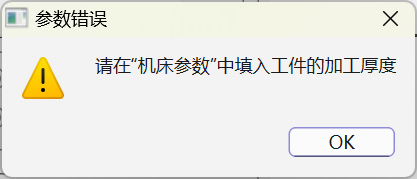


图 4-19加工厚度非法报警

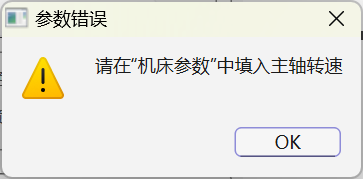


图 4-20主轴转速非法报警

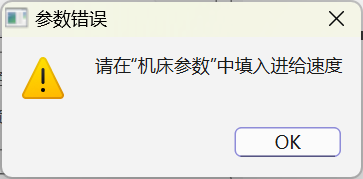


图 4-21进给速度非法报警

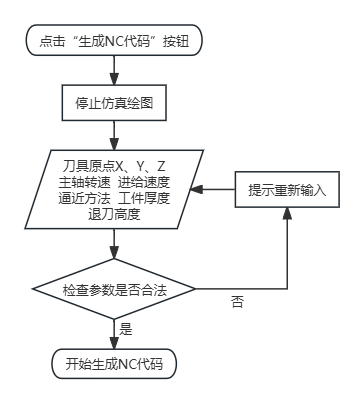


图 4-22点击“开始生成NC代码”按钮

如图 4-22所示，在点击生成NC代码按钮之后，程序将会先检查输入的各种参数，如果有非法参数，则会弹出图 4-18到图 4-20所示的警示窗口，避免非法参数的录入。在通过所有参数检查之后，才进入NC代码生成函数。

在检查参数之前，还有一个将仿真绘图停止的逻辑，这是为了防止仿真绘图和计算NC代码这两个高占用函数同时运行，使程序卡死。

#### NC代码开始部分

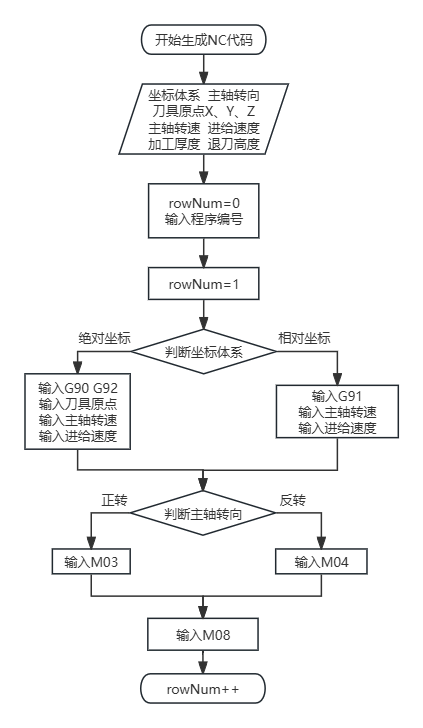


图 4-23 生成NC代码开始部分

NC代码的生成中，因为我们不需要编辑文本的样式，而且需要输出非常大量的文字，所以我们使用了相较于textEdit占用更少的plainTextEdit控件。并使用一个简单的rowNum变量来编写每行NC代码前的N指令。在点击按钮后，如图 4-23所示，在NC代码的第一行中，我们判断了坐标体系和主轴转向，并根据判断结果分别写上G90 G92或G91以及M03或M04。当选择绝对坐标时，我们还会传入刀具原位的三个位置参数，写在G92后面。主轴转速和进给速度的数据直接从界面上传入即可。在最后加上M08打开冷却液，第一行代码就结束了。

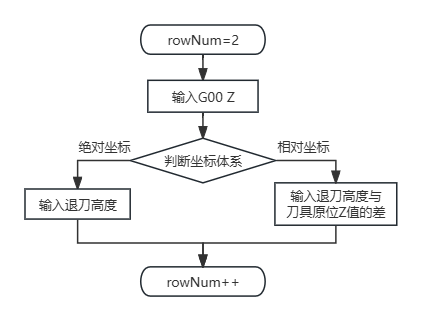


图 4-24 NC代码第二行

在第二行中，我们只执行一个退刀的动作，保证在刀具横向移动之前先退到安全高度，避免在横移时碰到加工毛坯而断刀，如图 4-24。

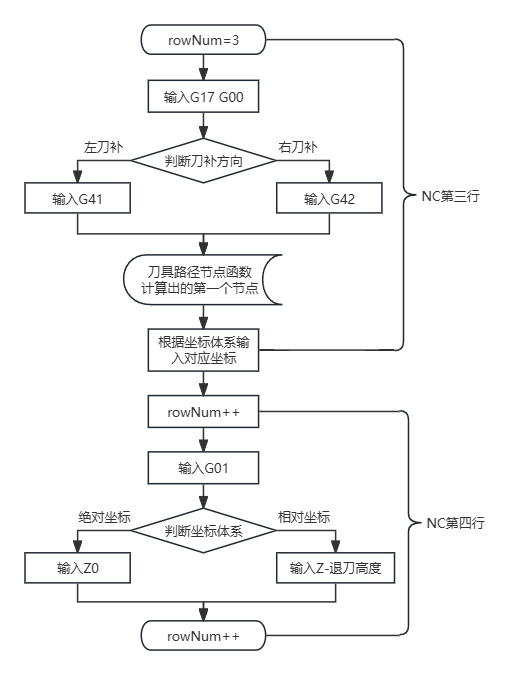


图 4-25相对坐标退刀

在第三行中，程序将刀具快速定位到下刀点的上方，并根据选择的刀补方向添加G41或G42，如图 4-25。

在选择相对坐标时，我们创建了两个用于相对坐标之间相减的中间变量realx和realy。这两个变量在之后的中间部分也有运用。第三行的最后，我们加上了D01命令以让机床确定刀具号码。

第四行和第二行一样只有一个动作，就是下刀，选择绝对坐标时添加G01 Z0即可，选择相对坐标则是直接Z后面接负的退刀高度。

#### NC代码中间部分

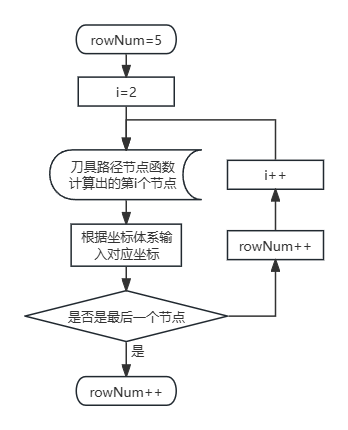


图 4-26 NC代码中间部分

中间部分逻辑非常简单，按照前面所述方法输出N指令，读取每个之前计算完成的坐标值并输出到plainTextEdit中即可。流程图如图 4-26。

#### NC代码结尾部分



图 4-27 NC代码结尾

如图 4-27，NC代码的结尾逻辑就是反过来的开始部分，倒数第三行进行退刀，倒数第二行回到刀具原点上方，并使用M05 M09关闭机床上的设备，最后一行使用M02结束程序即可。

### NC文件保存

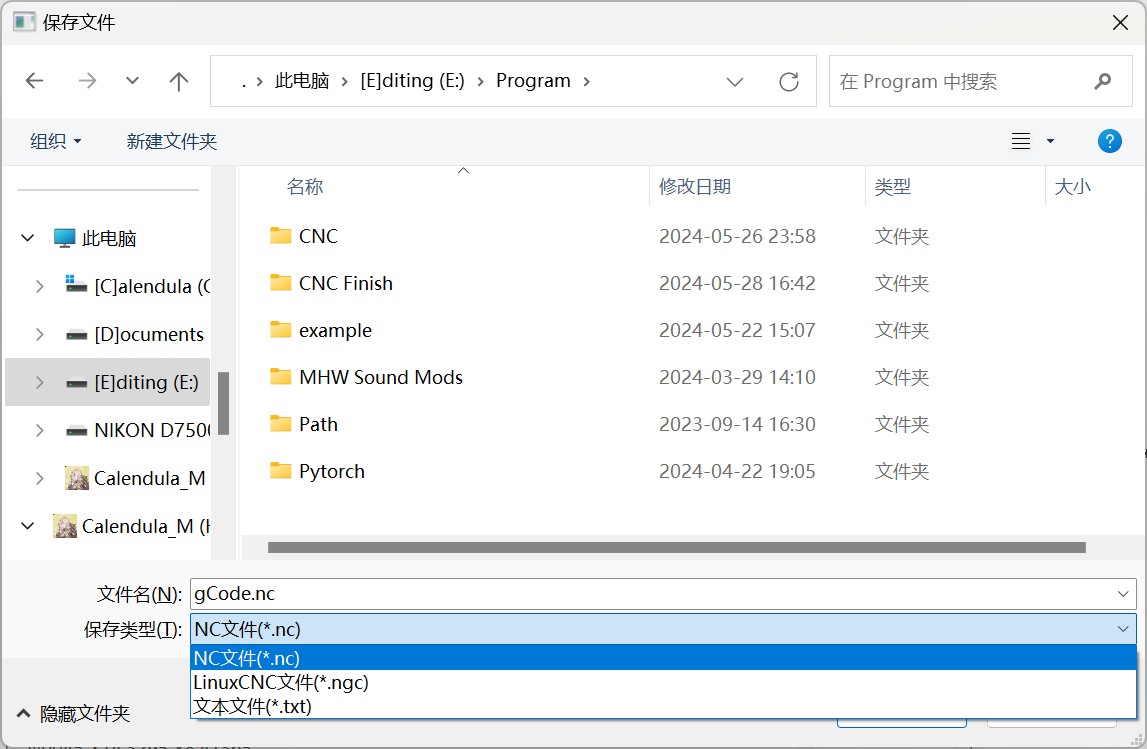


图 4-28文件保存界面

在点击“保存NC代码”按钮之后，程序会弹出如图 4-28所示的窗口来保存文件。文件默认名称为gCode.nc，用户可以选择以.nc/.ngc/.txt这三种常见的NC代码格式保存，保存时程序可以自动为文件补全文件后缀。

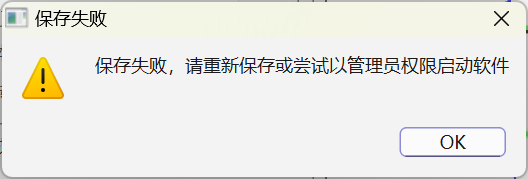


图 4-29保存失败

当程序无法创建文件、无法写入文件或用户主动关闭文件保存界面时，程序会弹出如图 4-29所示的警告窗口，提醒用户可能需要使用管理员权限打开程序。

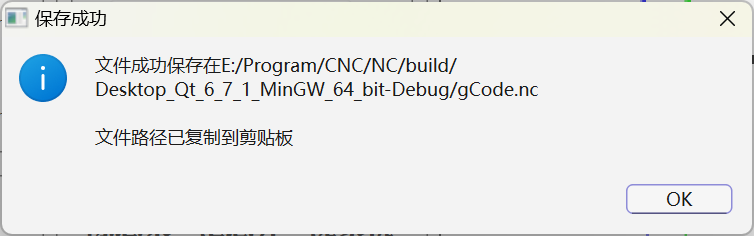


图 4-30保存成功

当文件保存成功之后，程序会弹出如图 4-30所示的提示窗口，告知用户文件的保存位置，并复制该位置到用户的剪贴板中，以防用户忘记保存位置且没有看弹窗上显示的保存位置而找不到刚刚保存好的文件。

#### 自动补全文件后缀

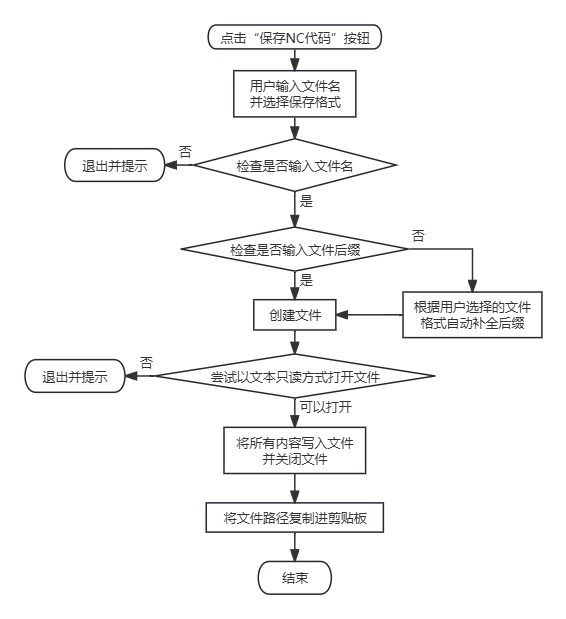


图 4-31自动补全后缀

从之前的叙述也能多少看出，我的程序设计逻辑非常偏向用户体验的舒适。所以在简单的保存之外，我还加入了保存路径选择，输入文件名、自动补全文件后缀等功能。图 4-31展示了保存文件的逻辑，这个逻辑可以让用户在选择与当前不同的格式保存时自动改变文件后缀名，而不用在窗口上重新输入。且当用户输入一个不带后缀的文件名时，自动根据用户当前选择的文件格式补全后缀名，大大增加了使用舒适度。

#### 文件保存及自动复制文件路径

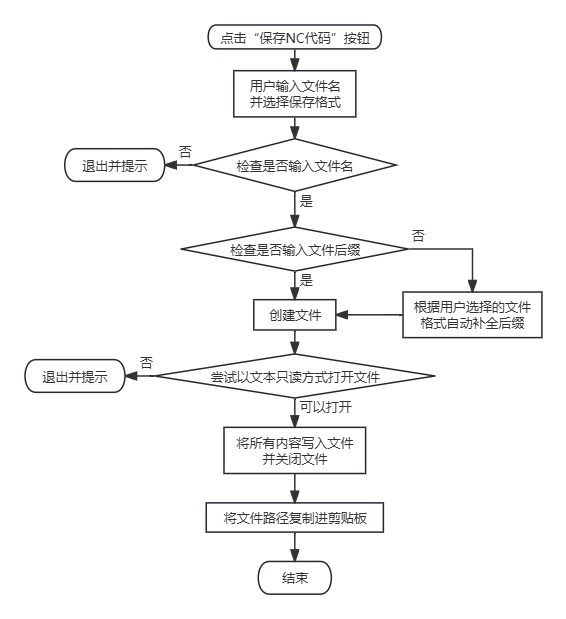


图 4-32保存文件

当程序获取了用户需要的文件名和保存路径之后，程序会尝试以可读可写的方式创建这个文件，并尝试以只读仅文本的方式打开这份文件，测试用户是否为文件设定了正确的文件格式。如果正确的话就将NC代码窗口中的内容以UTF8格式写入文件中，这是为了保证如果用户在编辑窗口加入了中文注释或程序名，程序也能正常写入文件。

将全部内容写入文件后，程序会关闭这个文件并用QClipBoard的功能把完整文件路径复制到剪贴板。

## 绘图区

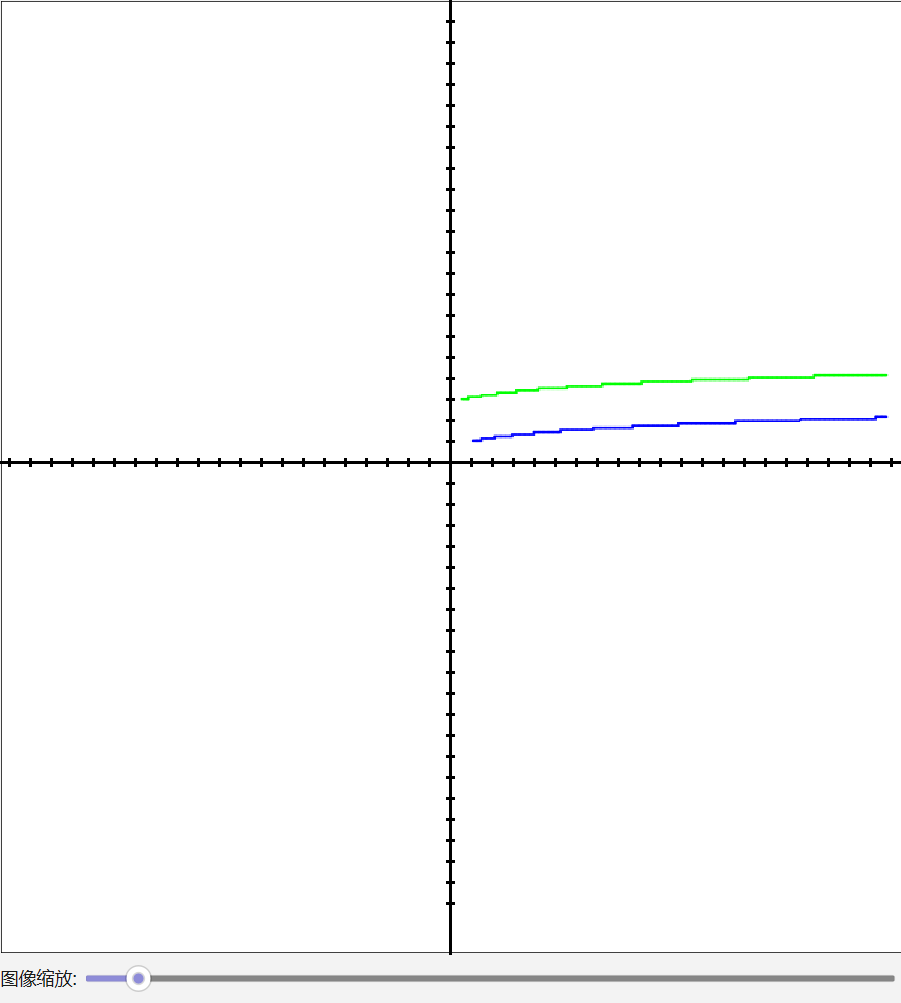


图 4-33绘图区域

如图 4-33，绘图区包含一个带直角坐标系的白底画布，和一个用来控制图像缩放的滑条。原曲线、刀具路径曲线和仿真时的刀具动画都会绘制在这个窗口。

### 窗口创建和坐标轴绘制

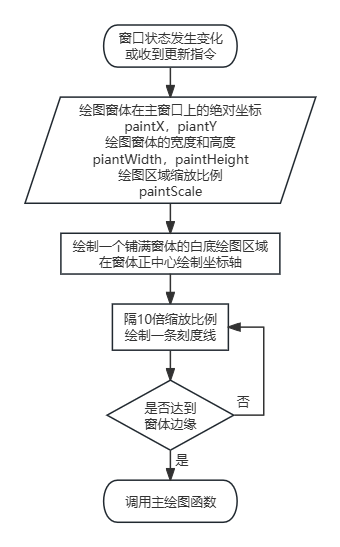


图 4-34窗口创建和坐标轴绘制

我们的窗口支持拖动改变大小，所以绘图窗口的大小也不是固定的，所以我们创建了一个widget对象，使用函数获取它的绝对坐标和区域的大小，以创建我们的绘图区域，如图 4-34所示。坐标轴上的刻度线之间的距离跟整个绘图区的缩放比例有关，这一点也可以很轻松地在for函数中控制i的步长来控制。由于窗口中的点是以一个像素为一个单位，显示出来之后会显得很小，我们确定了一个十倍的基础放大，这个系数在4.8.3中所述的主绘图函数中也有体现。在窗口的绘制事件最后，我们接入了主绘图函数，保证在窗口每次发生变化时主绘图函数的图像会跟随之变化。

### 绘图区域缩放

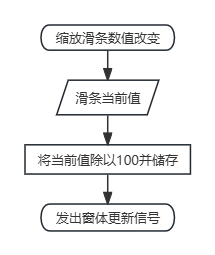


图 4-35绘图区域缩放

我们希望我们的绘图区域可以缩放10倍，但slider控件只能输出int值，为了保证滑条在滑动时的连续性，我们设置滑条的值范围为100到1000，步长为1，并在函数中将这个值除以100并赋给paintScale这个变量，如图 4-35所示。这样做我们就可以丝滑地控制绘图区图像的缩放。

### 主绘图函数

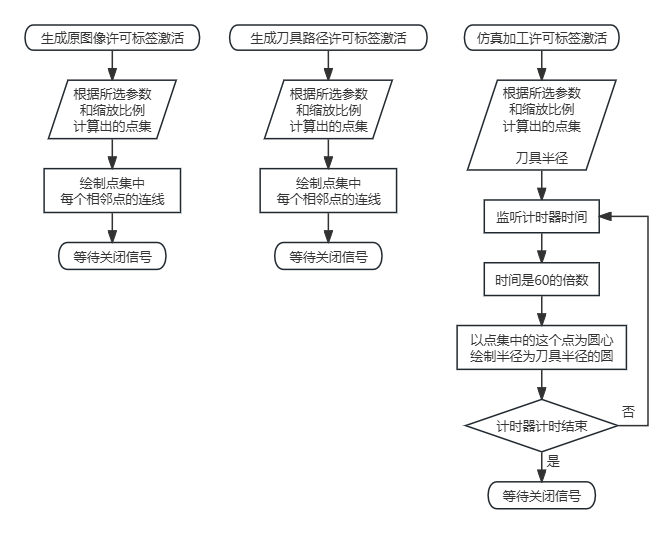


图 4-36主绘制函数

主绘图函数如图 4-36所示。当收到一个在下一节4.9中的函数生成的许可标签后，先设定好绘图区域和坐标系的变换，然后直接调用4.3.2中所述函数计算好的点进行绘图即可，原图像和刀具路径都是简单地将前后两个点用线相连，仿真动画则是在若干刀具点生成半径为刀具半径的圆形。其中原图像我们使用蓝色绘制，刀具路径使用绿色绘制，便于区分。仿真动画中我们使用了自带颜色库外的(255,200,200)这个较浅的红色，避免过多个表示刀具的圆形叠加在画面上时太过扎眼

坐标系的变换逻辑很好理解，先将坐标系原点移到绘图窗口正中心，在将Qt中向下的y轴变为向上即可，正好整个坐标系需要跟随图像缩放改变缩放比例，可以直接将坐标系乘上一个(paintScale,-paintScale)的变换。4.8.1中所述的这个十倍的基础放大也可以同时乘进这个变换中，但这样会使画出来的图线非常的粗，不是很好看，所以这个十倍的基础缩放被我们乘在了绘图行为的参数传入上。

## 主操作按钮区

图4-38 主操作按钮区

图 4-37主操作按钮区

如图 4-37所示，这个区域只包含了三个功能非常容易理解的按钮，用来发布三个bool型的许可标签：用来控制原路径绘图函数开关的shengchengyuanlujing、用来控制刀具路径绘图函数开关的shengchengdaojulujing和用来控制仿真动画绘图函数开关的kaishifangzhen。接下来是它们点击事件的具体逻辑：

### “生成/更新原始曲线”按钮

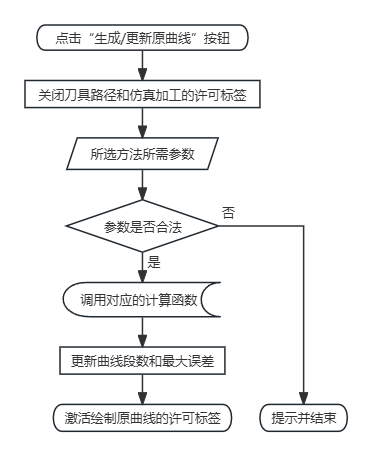


图 4-38生成、更新原始曲线按钮

图 4-38是点击“生成/更新原曲线”按钮后的逻辑，在用户点击按钮之后，程序先会将另外两个步骤的许可标签设为0，擦除与当前路径不相关的其他刀具路径和仿真动画。接着由于4.3.2中所述的逻辑，程序会先检查是否选择了逼近方法，接下来会分别检查4.2.2和4.2.3中所述的角度和幂数是否合法。这个幂数合法性检查其实只能检查用户是否输入了0或1这两个值，因为在窗口的初始化中已经限定了幂数输入框的上下限。

在参数合法性检查后，函数会调用计算路径点的函数来获取曲线的各个路径点，并获取这段路径的误差和段数，这条逻辑就是在写完4.3.2之后按照其中所述的逻辑加在这段函数中的。

最后，函数会将shengchengyuanlujing这个标签设为1，以让主绘图函数绘制原函数图像。

### “生成/更新刀具路径”按钮

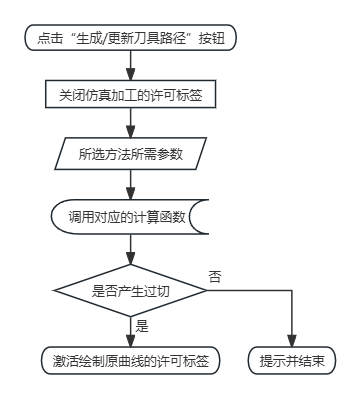


图 4-39生成、更新刀具路径按钮

“生成/更新刀具路径”按钮的逻辑中，由于写算法的同学写凸轮刀补函数时令函数返回值为检测是否过切，所以计算凸轮路径点的函数可以顺便放在判断条件中运行。当检测到过切时程序会发出过切警告并不发布生成刀具路径的标签，绘图函数也不会绘制该刀具路径。而对于幂函数，虽然不太可能会过切但同学也单独写了一个判断过切的函数，对此也只用在调用完计算函数之后写一个简单的逻辑判断即可，如图 4-40所示。

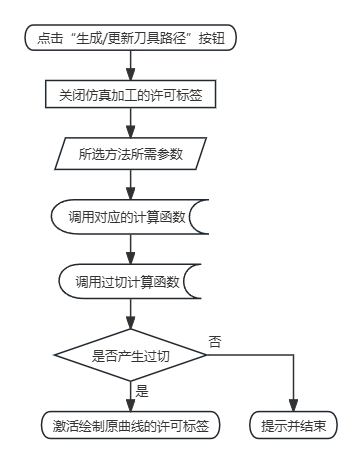


图 4-40幂函数过切判断

### “进行加工仿真”按钮

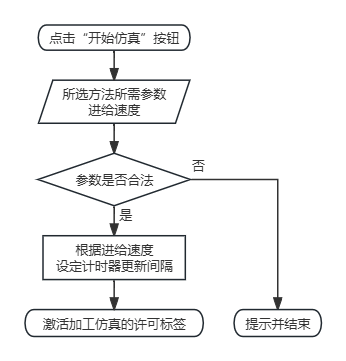


图 4-41点击“进行加工仿真”按钮

图 4-41是“进行加工仿真”按钮的逻辑，它在按下后会检查是否选择逼近方法、是否填入机床参数和刀具半径，并根据非法参数位置给出不一样的提示框。如果参数都合法，将会启动一个计时器，用于控制加工仿真的动画，其逻辑如图 4-44所示。

## 加工进度条

图4-43 加工进度条

图 4-42加工进度条

图 4-42是加工进度条，用于显示仿真加工的实时进度。

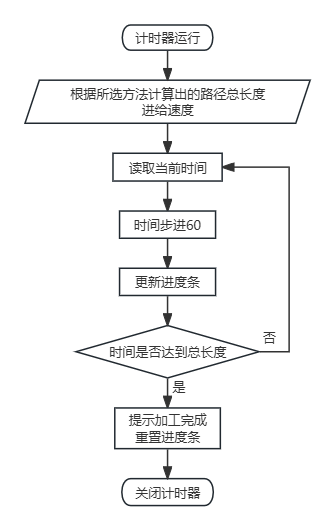


图 4-43仿真加工计时器

由于如果绘图窗口中有太多的圆形叠加，程序会变得非常卡顿，所以我们决定隔60个刀具点只绘制一个代表刀具的圆形，所以加工进度条也每60个计时器周期更新一次，逻辑也是十分简单的将当前步骤与所有步骤数相除再乘100以换算成百分制。

## UI设计补充

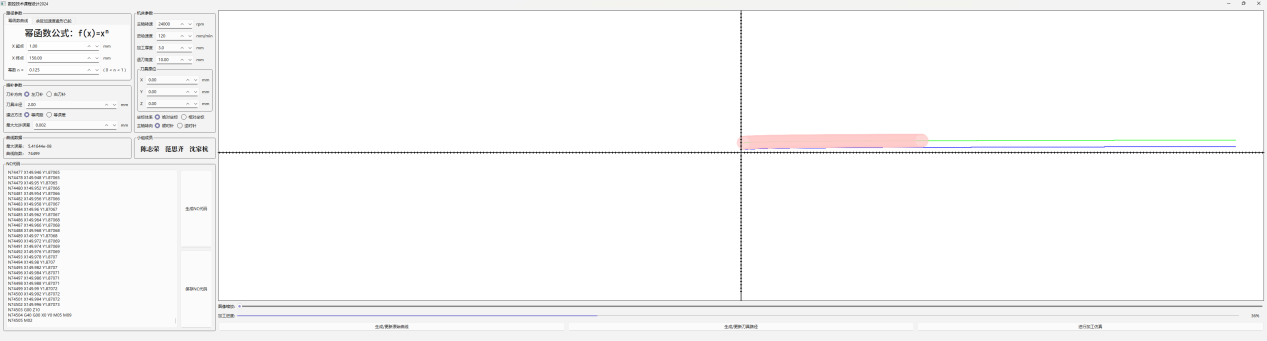


图 4-44程序在我的电脑上运行

在4.8.1中我提到了我们的软件支持变换窗口大小，对于一个需要找到某个不在主窗口正中心的窗口的正中心坐标，且还要根据这个中心点进行一系列坐标变换及图像绘制的程序，使用可变大小的窗口似乎不是一个很明智的选择，这会给UI的编程带来许多额外的计算量。但因为本人使用的显示器尺寸比较特殊，如图 4-44所示，这是我们制作的软件在我的屏幕上成功全屏运行的画面。因为这个特殊的尺寸，导致我经常在使用各种软件的时候因为其不适配自己的屏幕而产生烦恼，所以我在这次课设中给了自己一个挑战，反正UI部分是我一个人完成，那我就看看自己能否在一周的学习时间内，做到可以从零开始成功制作出自己满意的程序。最终，显然地，这个需求我较好地完成了。事实上，因为有了这个挑战，让我可以更深入地学习C++和Qt这个开发环境，我也在一步一步解决自己提出的问题中锻炼了根据问题独立分析独立思考的能力。

此外，在4.2.2中我提到了我们把幂函数的终点限制在200，是因为大多数屏幕都无法显示如此多的坐标，如图，这是在我这种极端尺寸下，最小缩放比例的，X终点为150的对应函数图像，终点为200时图像已经超出显示区域很大一部分了。

## 源码发布

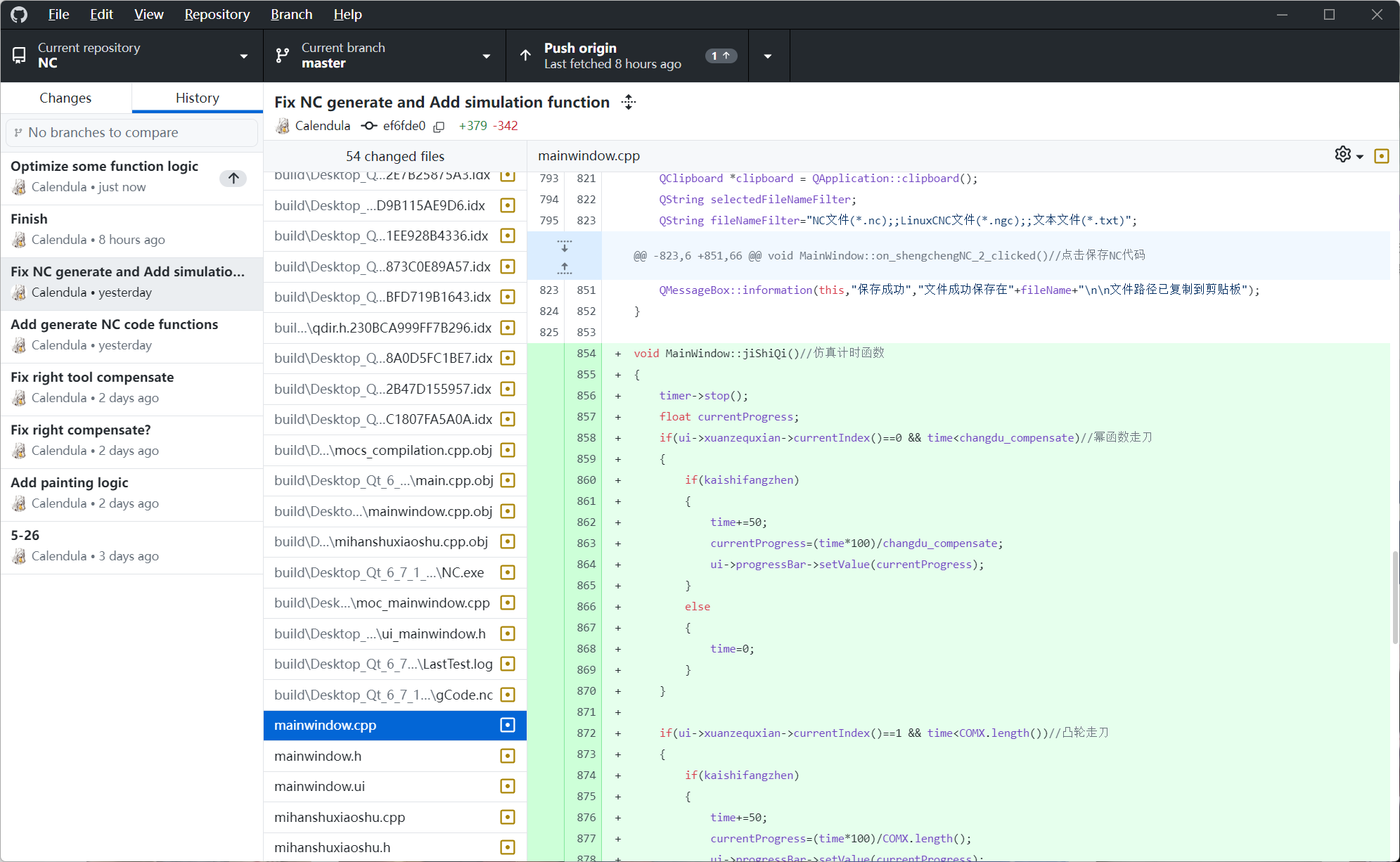


图 4-45将源码上传至GitHub

我个人是非常喜欢GitHub这个CodeChange视图的，它可以直观地让我看到我的代码在哪里有改动，我增删了多少东西。同时，班里许多同学都认为我有一定的编程基础，应该会做的比较好，所以经常来问我各种问题，我无法很快地向他们解释清楚诸如“这个NC代码的生成应该怎么写啊？”这种宽泛的问题。所以如图 4-45所示，我将我的整个工程文件夹全部上传至了GitHub托管。同学们问这种宽泛的问题时，我会让他们先去GitHub看我相对应模块的源码，有更细节的问题再与我讨论。这样我又不用浪费时间跟许多人反复解释同一个东西，也能在更细节的讨论交流中获得一些新的思路和想法。

本次作业也是有上一届的一位学长的指导，他向我分享了一些模块的编程思路，也帮我优化了许多模块的逻辑，非常感谢他。这份作业的仓库是https://github.com/Calendula47/NC，如果学弟学妹们也有幸看到这份报告，也可以去看看思路，希望能帮到大家。我的程序有非常多可以和需要改动的地方，我也在里面留了许多小彩蛋。

# 总结

在本次课程设计中，我们小组成功开发了一款面向非圆曲线的计算机辅助制造（CAM）软件。软件采用Qt作为界面开发平台，利用C++语言进行编程，实现了对非圆曲线如幂函数曲线和余弦加减速盘形凸轮廓线的精确逼近处理。通过等间距直线逼近、等弦长直线逼近和等误差直线逼近等算法，我们能够有效地生成曲线节点，并据此自动生成刀具中心轨迹和NC代码，同时提供了仿真加工过程的可视化展示。

**关键成果包括**：

**算法实现**：软件集成了多种逼近算法，包括等间距、等弦长和等误差直线逼近，以及它们的优化版本，确保了曲线逼近的高精度和效率。

**刀具补偿**：实现了刀具半径补偿功能，支持左右刀补，能够根据给定的补偿量和进给方向自动计算刀具中心轨迹，并通过过切报警功能增强了加工安全性。

**NC代码生成**：软件能够根据生成的轨迹自动生成NC代码，支持保存为文本文件，便于后续的加工应用。

**仿真加工**：提供了仿真加工功能，用户可以在屏幕上直观地观察刀具按照设定参数进行加工的模拟过程。

**用户界面**：设计了友好的用户界面，包括路径参数设置、插补参数设置、曲线数据显示、机床参数设置等功能区域，提高了软件的易用性和交互性。

**软件测试**：通过多个实例验证了软件的各项功能，确保了软件的稳定性和可靠性。

**学习与提升**：

**技术能力**：小组成员通过本次设计，加深了对数控加工技术、软件开发和用户界面设计的理解，提升了编程和软件设计能力。

**团队协作**：在共同完成项目的过程中，团队成员间的良好协作对于项目成功起到了关键作用。

**问题解决**：面对开发中遇到的技术难题，我们通过查阅资料、集思广益，独立分析和解决了问题。

**展望未来**：

**功能扩展**：软件未来可以进一步扩展更多类型的非圆曲线逼近算法，增加更多的刀具补偿方式，以及更丰富的仿真加工功能。

**性能优化**：对现有算法进行性能优化，提高计算速度和软件的响应效率。

**用户体验**：持续改进用户界面和交互逻辑，提供更加舒适和直观的用户体验。

通过本次课程设计，我们不仅加深了对非圆曲线CAM软件开发的理解，而且在实际编程实践中提高了我们的技能。软件的成功实现证明了我们在算法设计、软件架构设计、用户界面设计以及数控加工模拟方面的综合能力。我们的软件展示了幂函数曲线和余弦加减速盘形凸轮廓线的自动编程和仿真加工过程，不仅提升了我们编程和软件开发技能，还深入理解了数控加工技术和算法实现，为未来的学习和工作打下了坚实的基础。