基于嵌入式系统的运动控制器设计
课程实习报告

学　　院：自动化学院

指导老师：李勇波

学　　号：20201000128

班　　级：231202

姓　　名：刘瑾瑾

小组成员：刘瑾瑾、程婷婷、杨文丽

二〇二三年二月

目录

目录

第一章 绪论

1.1 设计要求

利用实验室的平面二维控制平台，通过嵌入式开发设备发出的指令，给予平台上两个电机驱动器相应的控制信号，控制两台电机的转动方向和速度，从而控制两轴交叉部位末端执行点的移动，从而在二维平面上进行绘图。需要以下功能：

1.基础功能：

(1) 四象限直线插补;

(2) 四象限圆弧插补;

(3) LCD 屏动态实时坐标显示;

(4) 八段数码管显示;

(5) 键盘扫描输入;

(6) 串口通信;

2.优化功能：

(1) 顺时针画圆；逆时针画圆;

(2) X 轴、Y 轴正方向手动加减速控制;

(3) XY 等速斜向运动;

(4) 给定圆心和半径画任意圆;

(5) 画菱形、五角形、圆和圆锥;

(6) 正反向画创意设计图形小车;

(7) 串口通信下发指令画五角形。

1.2 实验环节与设备介绍

平面二维控制平台由两台伺服电机、两根以螺旋转轴和轴承为基础的滑轨、两个伺服电机驱动模块、以及供电模块、急停模块等装置组成。

控制上选用的是上学期电路综合实习我们自己所焊接的开发板，具有独立键盘、矩阵键盘、共阴极数码管、LCD 显示屏、LED 灯、EEPROM 存储记忆和 ADDA 模块等多种模块，控制电路由显示电路、键盘电路、脉冲锁存和光槽开关信号读取电路、光隔电路组成。

两台伺服电机各连接一根螺旋转轴，两者分别作为平面二维控制系统的 X 轴与 Y 轴，当伺服电机顺时针旋转时（即点击接受正向脉冲信号），交点将向远离平台的方向移动。在 X 轴与 Y 轴的交点处向实验台外延伸出

一根铝管，在铝管头夹上一支笔，就可以利用笔在平面的纸张上进行图案的绘制，从而验证控制正确性与精度。

1.3 整体工作介绍

本组可通过按键点动控制 X 轴、Y 轴正反向运动，进行加速和减速控制，完成了基础的直线和圆弧插补绘画工作，实现运行坐标的实时显示，并结合开发板的现有模块增加了一些创新性功能，例如圆弧的顺时针和逆时针绘制以及实现复杂组合系统小车的正反画。

在绘制过程中，伺服控制系统运行流畅，可自由调节 X 轴和 Y 轴的运行方向和速度，自由设置画图的起始位置和图形的大小以及形状，如：菱形、五角形，通过设置坐标点的方式组合直线和圆弧可得到复杂图形，如：圆锥和小车。

总而言之，在本组成员的共同努力下，较好地完成了所有设计要求，并优化了部分功能，具有一定的创新性。

第二章 插补算法原理

插补计算就是对数控系统输入基本数据(如直线的起点、终点坐标，圆弧的起点、终点、圆心坐标等)，运用一定的算法计算，根据计算结果向相应的坐标发出进给指令。对应着每一进给指令，机床在相应的坐标方向上移动一定的距离。

2.1 直线插补算法

2.1.1 算法原理

加工如图 2.1 所示的平面斜线 AB，以斜线起点 A 的坐标为 X_0，Y_0，斜线 AB 的终点坐标为(X_e, Y_e)，则此直线方程为：

$$\frac{X-X0}{Y-Y0}=\frac{Xe-X0}{Ye-Y0}$$

取判别函数 $F=(Y-Y_0)(X_e-X_o)-(X-X_0)(Y_e-Y_0)$

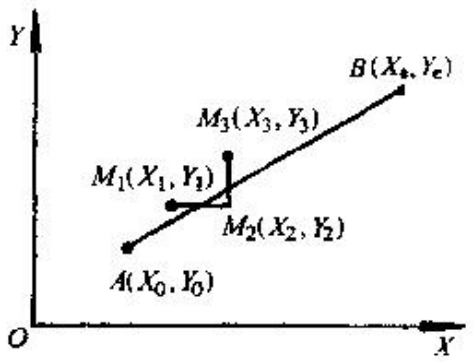
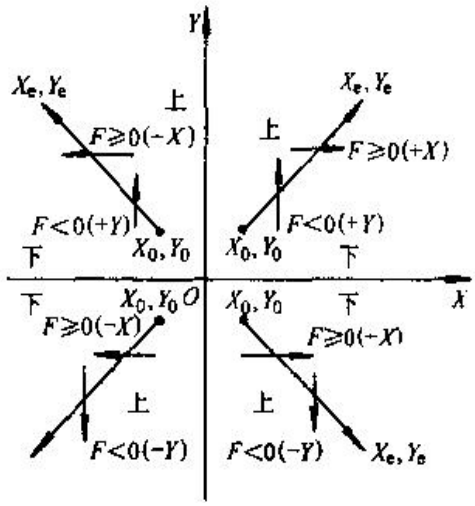


图 2.1 直线插补算法示意图

在第一象限插补时，若沿 x 方向走一步即 $X_{i+1}=X_i+1$ 则 $F_{i+1}=F_i-(Y_e-Y_0)$，若沿 Y 方向走一步，即 $Y_{i+1}=Y_i+1$ 则 $Fi+1=Fi+(X_e-X_0)$。偏差值 F_{i+1} 的计算只用到前一点的偏



差值 F_i 和斜线的长度在坐标方向的投影(X_e-X_0)，(Y_e-Y_0)。

图 2.2 直线插补四象限示意图

图 2.2 是象限的划分规则，根据对线段加工方向的不同来判别它所处的象限，对于四个象限可共用如下列的判别式：

向 X_e 方向走一步 $F_{i+1}=F_i-|(Y_e-Y_0)|$;

向 Y_e 方向走一步 $F_{i+1}=F_i+|(X_e-X_0)|$;

移动方向的控制可根据线段所处的象限来决定，四象限的移动方向如下所示：

表 2.1 象限判别和电机方向

方向	第一象限	第二象限	第三象限	第四象限
Xe-X0	>0	<0	<0	>0
Ye-Y0	>0	>0	<0	<0
X 向电机	正	反	反	正
Y 向电机	正	正	反	反

直线插补的终点判断：把被加工线段的 X_e-X_0，Y_e-Y_0 的长度单位换算成脉冲数值(若长度单位为 mm，则把上述的坐标增量值除以脉冲当量),然后求出各坐标方向所需的脉冲数总和 n：即 $n=|(X_e-X_0)|+|(Y_e-Y_0)|$，计算机无论向哪个方向输出一个脉冲都作 n-1 计算，直到 n=0 为止。

2.1.2 算法实现流程

进入直线插补程序，首先对相关参数进行初始化：$|X_e-X_o|$,$|Y_e-Y_o|$，计算 $n=|(X_e-X_o)|+|(Y_e-Y_o)|$,通过判别函数 $F=(Y-Y_0)(X_e-X_o)-(X-X_0)(Y_e-Y_0)$ 的符号判断沿 Xe 方向移动或沿 Ye 方向移动，移动后，重新计算判别函数，且 n=n-1，若 n=0,则直线插补结束，否则继续重复上述步骤。

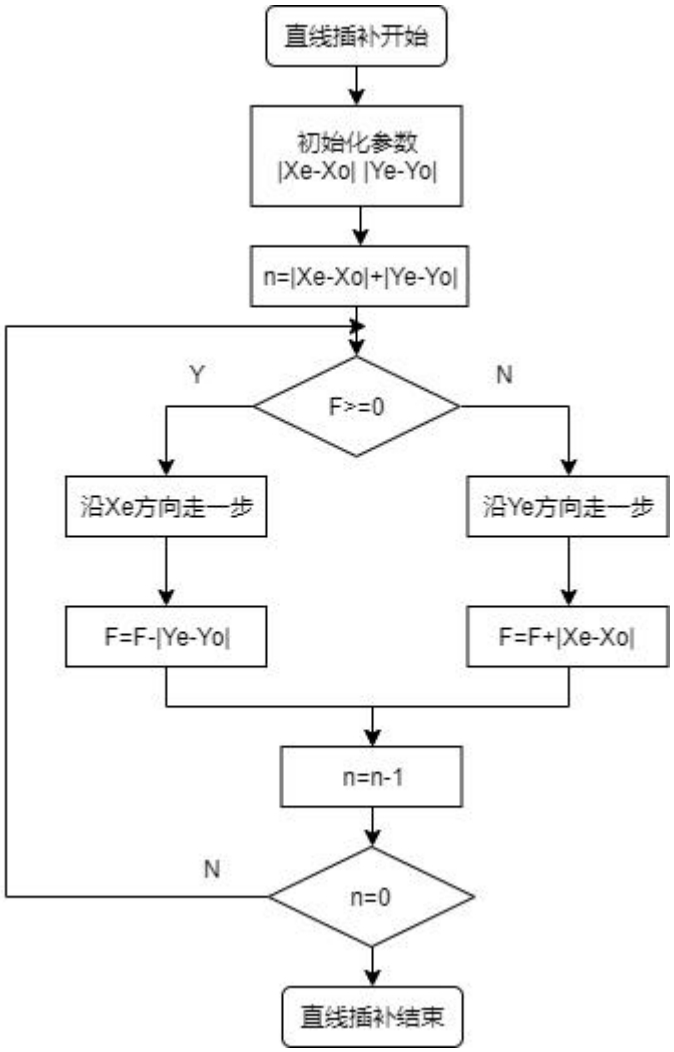


图 2.3 直线插补流程图

2.2 圆弧插补算法

2.2.1 算法原理

在图 2.3 中 AB 是被加工圆弧。加工程序中给出的已知条件通常是 A 点 B 点的坐标值，圆心 O'点相对圆弧起点 A 的增量坐标值。由图可知：圆心 O'点相对 A 点的增量坐标值为$(-I_o，-J_o)$。改变符号后就成为 A 点相对 O'点的增量值 I_o，J_o。由此可求出圆弧的半径值 R：$R^2=I_o^2+J_o^2$ 在以圆心 O'点为原点的 I、J 坐标系中，圆的方程可表示为：$I^2+J^2=R^2$。设刀具已位于 M_1 点，则 M_i 点对圆弧 AB 的位置有三种情况：

1) Mi 在圆弧外侧，则 O' $M_i>R$，$I_i^2+J_i^2>R^2$

2) Mi 在圆弧上，则 O' $M_i=R$，$I_i^2+J_i^2=R^2$

3) Mi 在圆弧内侧，O' $M_i<R$，$I_i^2+J_i^2<R^2$

Mi 点的判别式可写成如下形式：$F_i=I_i^2+J_i^2-R^2$。

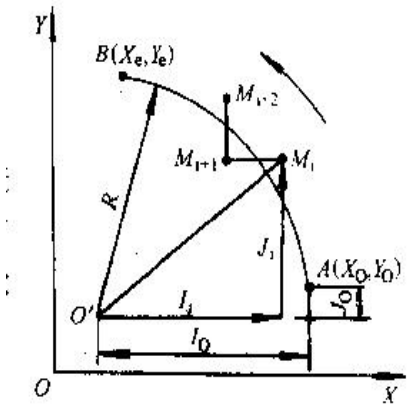


图 2.4 圆弧插补

上述为在第一象限逆时针加工圆弧(称逆圆弧)的情况，而在第一象陨顺时针加工圆弧(顺圆弧)和第二、三、四象限加工顺圆弧和逆圆弧时，判别式都不相同。带符号运算时，无论在哪个象限工作，顺圆弧或逆圆弧，归纳起来有如下四种情况：

1）+X 方向走一步

2）-X 方向走一步

3）+Y 方向走一步

4）-Y 方向走一步

将四象限规律总结为下表所示：

表 2.2 象限判断和电机转向

		第一象限	第二象限	第三象限	第四象限
Ii 的符号		+	−	−	+
Ji 的符号		+	+	−	−
X 向电机	顺圆	+	+	−	−
	逆圆	−	−	+	+
Y 向电机	顺圆	−	+	+	−
	逆圆	+	−	−	+

2.2.2 算法实现流程

由表 2.2 知，使用逐点比较法无论是在四个象限画圆或是正反画圆，圆心的符号和 X、Y 方向电机的旋转方向都不同，为此我们需要设计多个画圆程序，在每个象限都要单独调用对应象限的程序，但这样函数的调用会比较复杂，且容易出错。因此，我们将程序进行整合，通过分支结构实现任意画圆程序，该程序可以实现在任意象限画圆，且，能够在给出圆心坐标、半径、目标位置坐标以及顺逆时针后就能画出需要的圆，这也为后续实现复杂组合图形的正画和反画奠定了基础。

首先，判断当前点和目标点是否在同一个象限内，如果在同一个象限则

直接调用所在象限的画圆程序；否则，则根据顺时针和逆时针以及当前象限，调用当前象限对应的程序画到坐标轴，直到到达同一象限为止。流程图如图2.5 所示：

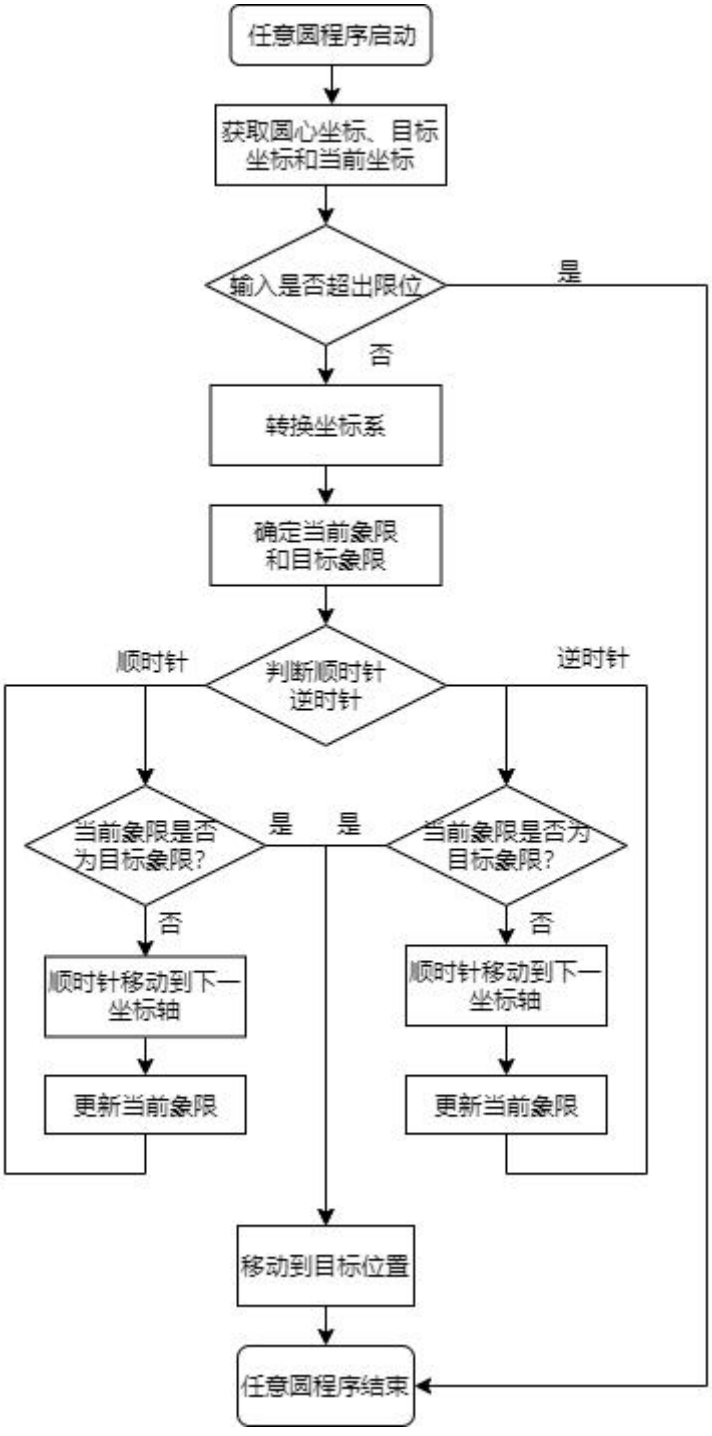


图 2. 5 任意圆程序流程图

第三章 系统搭建

整个系统以 STC89C52RC 单片机作为核心控制伺服电机运动，由键盘作为输入模块进行功能选择，LCD 显示屏显示实时坐标且进行功能选择提示，共阴极八段数码管进行坐标和半径等相关参数的输入，且上位机可通过串口通信向单片机发送指令。

3.1 输入部分模块选择与设计

3.1.1 矩阵键盘模块

输入部分，本组选择使用矩阵式键盘模块。其电路图如图 3.1 所示。

矩阵键盘大小为 4x4，共 16 个键位的矩阵键盘模块，其电路如图 3.1 所示。该键盘可以利用行扫描和列扫描的方式，确定矩阵键盘上的哪一个按键被按下。列扫描时，首先将接在列上的所有 IO 口置高电平，接在行上的所有 IO 口置低电平。当其中有一列的任何一个按键被按下时，整条列线都会因此而被拉低，变为低电平。反之，进行行扫描时，先将接在行上的所有 IO 口置高电平，接在列上的所有 IO 口置低电平。当其中有一行的任何一个按键被按下时，整条列线都会因此而被拉低，变为低电平。由此确定了被按下按键的行号和列号，就可以用来确定按键的位置，从而执行相应的功能。

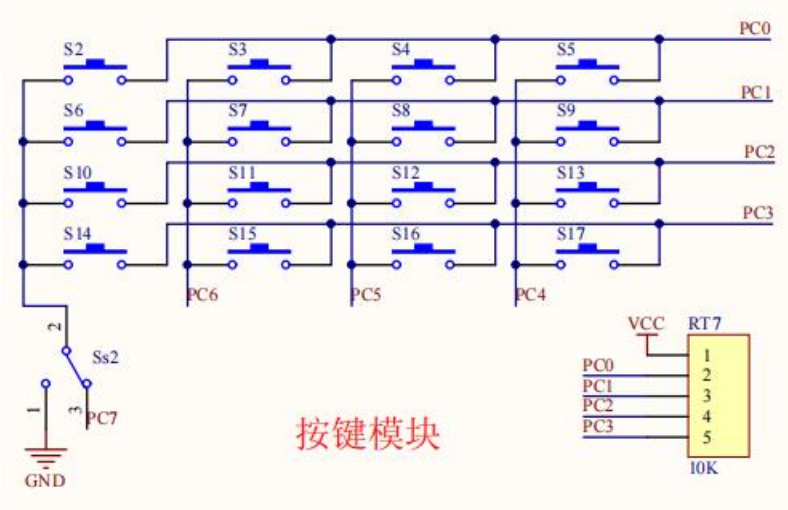


图 3.1 按键模块电路图

3.1.2 矩阵键盘功能选择设计

由于包含功能较多，如：X、Y 轴正反向运动，加速、减速，以及画圆、圆锥和五角形等，故我们对每个按键控制的功能进行了具体设计，如图 3.2 所示：

清零/暂停/复位	X正向运动	圆锥	指定圆心和半径后画整圆
XY等速斜向运动	X负向运动	菱形	画五角星
加速	Y正向运动	暂无功能	正向小车
减速	Y负向运动	暂无功能	反向小车

图 3.2 矩阵键盘功能设计图

当需要输入坐标或者数字时，键盘功能对应表如下：

0	4	8	未定义
·1	5	9	未定义
2	6	未定义	未定义
3	7	未定义	未定义

图 3.3 矩阵键盘数字表

3.2 输出部分模块选择与设计

3.2.1 LCD 显示屏

为方便使用者进行功能选择和观察实验仪当前运动位置，我们使用字库在 LCD 显示屏上进行当前功能选择和实时坐标显示，实现数据可视化。液晶屏可显示 32 个字符，包括标点符号、英文大小写和阿拉伯数字等。

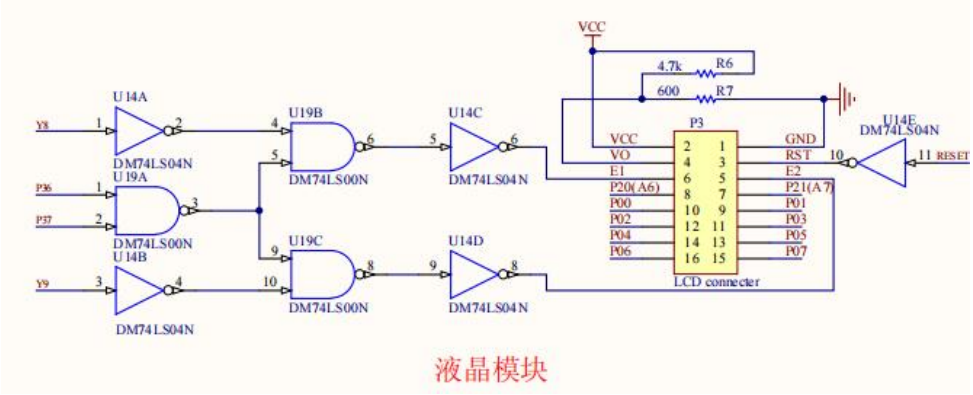


图 3.4 LCD 显示屏电路图

在 LCD 界面中，会显示当前的 X、Y 轴所处坐标（位数为 4 位，默认为 0),

以及用户可以通过矩阵键盘可以操作的内容，包括指定画圆、画五角形等。

在具有输入功能模块时，LCD 界面提示输入的内容，参照上一个模块的键盘表按键输入即可。具体显示如下图所示：

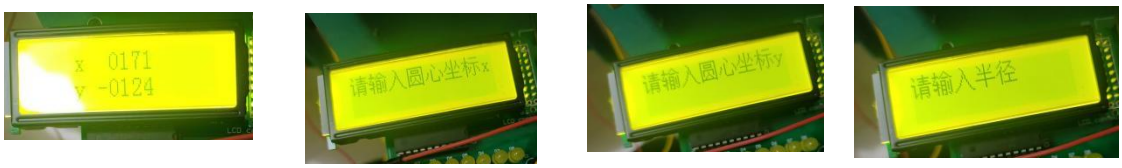


图 3.5 LCD 显示示例

3.2.2 8 段共阴极数码管

程序中有一些参数需要用户自定义输入，而如果使用 LCD 显示屏实现数据输入，则需对数据进行拆分重新设置字库，增加了程序在界面部分占据的内存，对程序运行效率也有一定的影响。故使用 8 段共阴极数码管实现输入数据的可视化显示，降低程序的复杂度，增加对相关模块的使用。

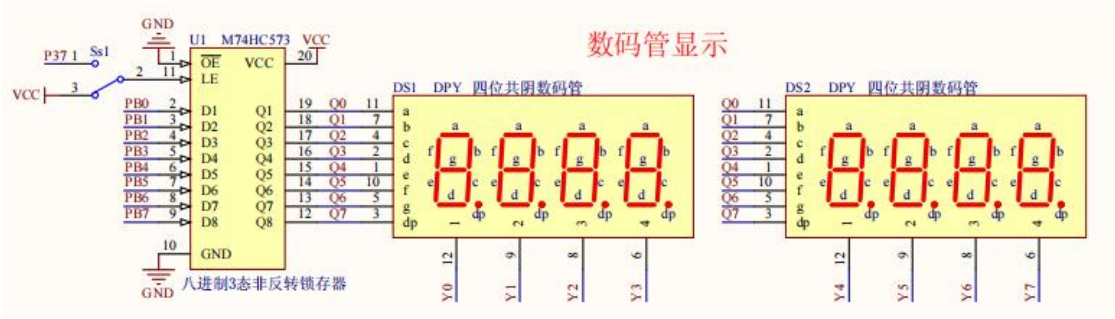


图 3.6 共阴极数码管电路原理图

当使用参数输入的功能时，通过图 3.3 所示对应按键输入数字 0-9，对 4 位数字依次输入。在数据输入过程中数据可以自动左移，列入输入数字 1234，则按下 1 键后数码管最右端显示 1，按下 2 键后 2 占据原来最右端的数码管，1 要左移一位。依次类推，显示内容如下：0001->0012->0123->1234。4 位数字输入结束后，可按任意键结束。

例如：设置半径为 2000。如图 3.6 所示：

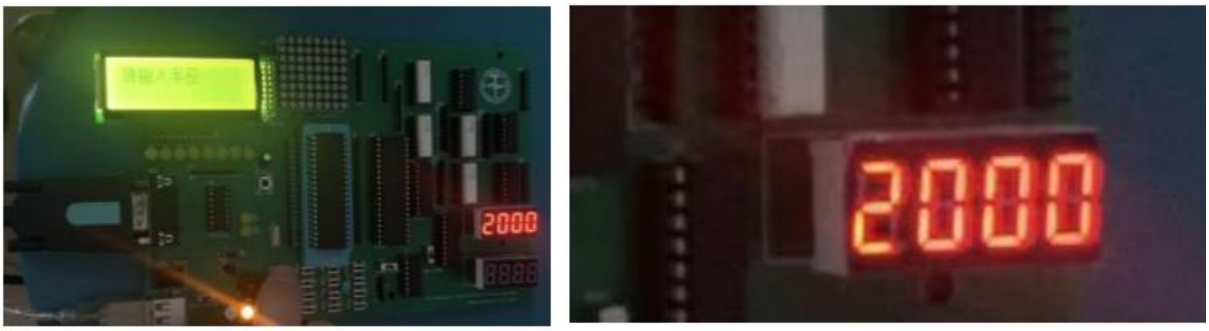


图 3.7 数码管输入示例

3.3 系统重点优化设计

3.3.1 显示参数刷新问题

在本系统中，我们组实现了 XY 实时坐标显示，增强了数据的可视化程度。但每次 XY 坐标更新后，LCD 显示屏上的数字都需要实时更新显示，这极大的占用了单片机的内部系统主控资源，影响单片机中主程序的运行，造成整个系统卡顿甚至无法正常执行操作，表现在二维运动平台上则为电机行进速度缓慢以及运行停止。

所以，要解决该问题就需要降低 LCD 显示屏的刷新速率，使其占用相对小的实时资源，但不可过低，过低会影响数据的实时性。因此我们采用每隔 4 个定时\计数器周期更新一次的方法，既降低对系统资源的占用，又在一定程度上保证数据的实时性，通过视觉暂留效应方便用户观察数据。这样的处理方法既解决了程序自身的卡顿问题，又给予用户较好的交互体验，是本系统设计的亮点之一。

3.3.2 LCD 字库内存问题

LCD 需显示较多信息，但该程序的储存空间不足会导致部分代码被搬移到 EPROM 从而无法正常运行，且大部分功能显示只有个别改动，不需要重新创建字库，因此我们采用覆盖结合清屏函数 clear()，并不对所有显示进行改动，而是必要时才清屏，其他时间采用直接覆盖。这样使代码的容量大大降低，优化节省空间的同时节省了运行时间，提高了程序的运行效率，也是本系统设计的亮点之一。

第四章 相关功能设计

4.1 全功能流程图

本次二维运动控制系统设计利用了实习板的按键模块、LCD 显示模块、8 段共阴极数码管模块等，通过编写调试程序控制二维运动控制实验装置进行图像的绘制。如图 4.1 为本系统的全功能流程图，由于内存空间限制，最终程序包含以下功能：

(1) XY 轴正反向运动及加减速

(2) XY 等速斜向运动

(3) 画菱形

(4) 通过按键和串口指令画五角星

(5) 画指定圆心和半径的圆

(6) 正向画小车

(7) 反向画小车

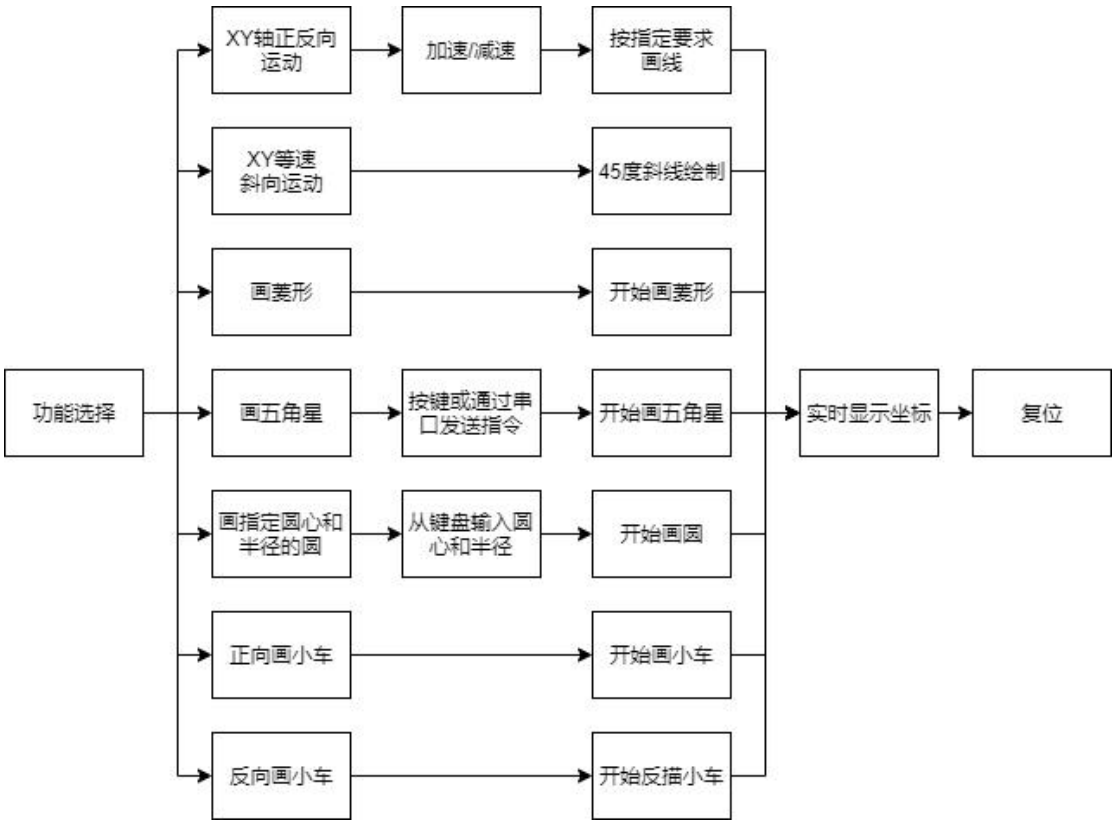


图 4.1 系统全功能流程图

4.2 画菱形部分

给定坐标，画长度相等的四条直线构成菱形，此功能主要用来验证四象限直线插补算法的正常运行，按键盘左上角复位键即可退出该程序。实际运行结果如图 4.2 所示：

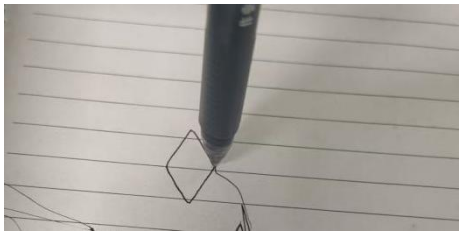


图 4.2 菱形演示图

4.3 画五角星部分

相比于菱形坐标，五角星的坐标计算更复杂一点，设五角星边长为 a,如图 4.3 所示，五角星五个坐标点公式如下：

A（a+a*sin18°，0），B（acos36°,-acos36°*cot18°+acos18°）

C（-acos36°,-acos36°*cot18°+acos18°），D(-a-a*sin18°,0) E(0,acos18°)。

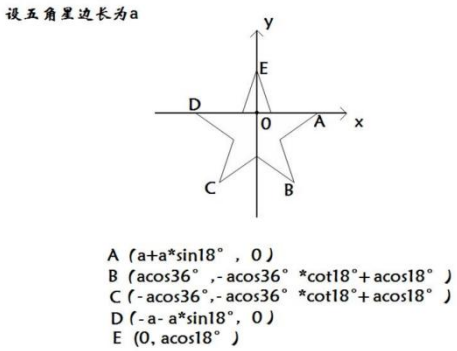


图 4.3 五角星坐标求解

点击键盘对应按钮可直接进入画五角星功能，或者通过串口助手发送指令‘a’，也可进行五角星的绘制，按键盘左上角复位键即可退出该程序。串口配置和实际运行结果分别如图 4.4 和图 4.5 所示：

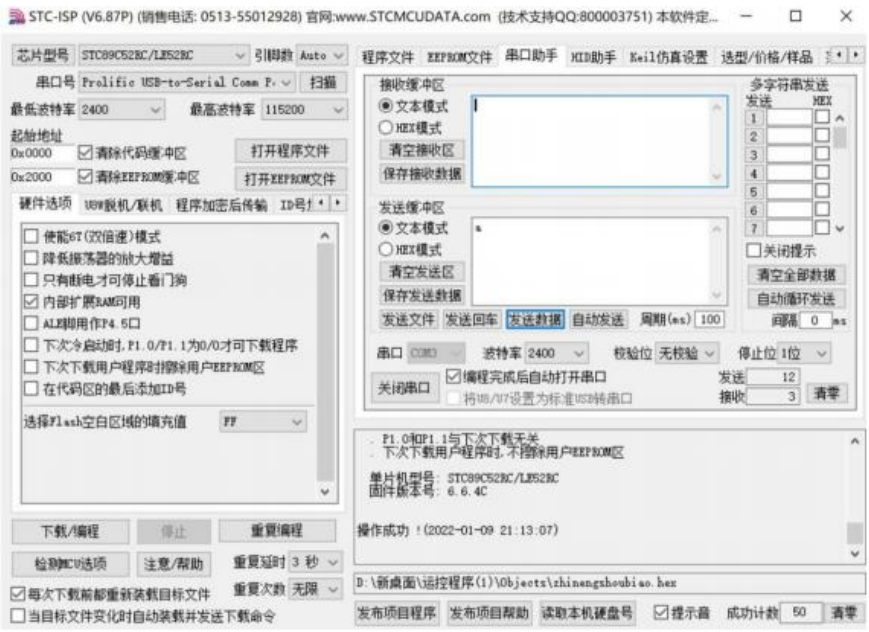


图 4.4 串口配置图

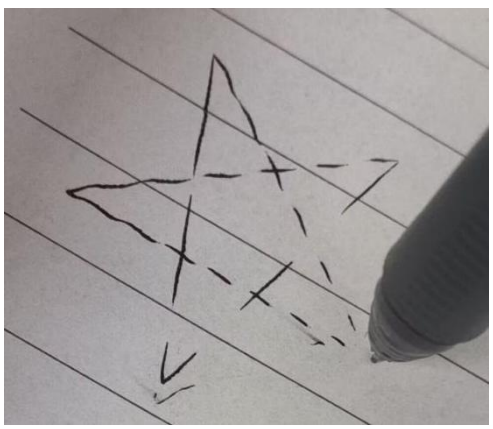


图 4.5 五角星演示图

4.4 画圆部分

点击键盘对应按钮，进入画圆模式。此模式下 LCD 会显示“请输入圆心坐标 X”，点击矩阵键盘对应数字输入按钮，4 位数字输入完毕后按任意键，LCD 显示“请输入圆心坐标 Y”，同样按任意键，显示“请输入半径”，输入完成，按任意键开始画圆。可得到想要的圆。按键盘左上角复位键可退出该程序。实际运行结果如图 4.6 所示。

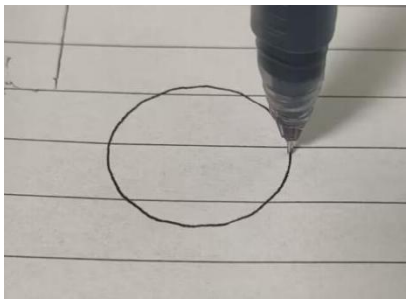


图 4.6 圆演示图

4.5 画圆锥部分

由于我们对画任意圆程序进行过组合优化，只需给出圆心坐标、终点坐标、半径和绘制方向（顺时针、逆时针），即可绘制对应的圆弧。通过直线和圆弧的组合，设置好交点坐标，即可实现对圆锥的绘制。点击相应按钮，进入画圆锥的模式，绘制完成后，按复位键退出该功能。实际运行结果如图 4.7 所示。

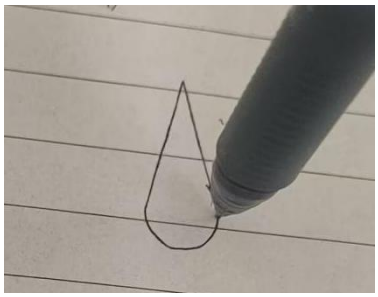


图 4.7 圆锥演示图

4.6 画小车部分

点击键盘对应功能按钮，进入画小车模式。小车属于复杂组合图形，但本质还是由直线和圆弧构成。将图像分解为圆弧和直线的组合，进行适当的比例优化，确定关键参数即可绘制小车。值得注意的是，正方向绘制时顺序和方向会有变化，编写程序时应当注意前后顺序变化。按键盘左上角复位键即可退出程序，实际运行结果如图 4.8 所示。

当然，绘制的组合图形不仅仅局限于小车，小车只是一个具体的体现。只要是可以用直线和圆弧组合而成的图形，都可以通过确定关键参数绘制。

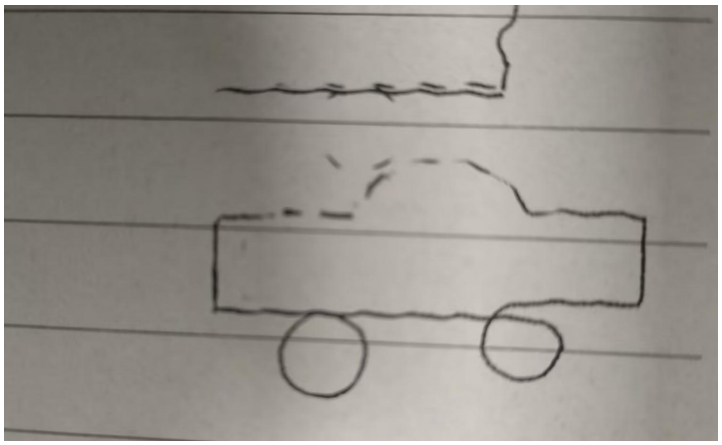


图 4.8 小车演示图

第五章 个人工作描述与实习总结

在本次基于嵌入式系统的运动控制设计中，我主要负责直线插补、圆弧插补的算法设计以及最终部分图形的绘制设计，部分参与输入输出模块选择、矩阵键盘排布与编码、LCD 显示控制、8 位数码管刷新显示控制、全系统整合和设计、显示界面设计与实现、各功能定义与测试以及系统整体代码运行与调试等，与队友合作共同解决问题。

在进行直线插补算法的设计中，参照实验指导书的流程图能够较快完成。而圆弧插补由于各象限以及顺逆时针都会对 XY 轴的运动方向有影响，所以我首先编写了各象限画圆弧的子函数，然后通过 switch 分支语句将子函数整合编写为任意圆程序，给出圆心、终点坐标、半径和方向即可画出指定圆弧，方便画图时调用。

通过本次实习，我对嵌入式系统开发设计有了更加深刻的理解，对于伺服电机的运动控制也更加清楚。在进行算法编写时，我深刻认识到，编写的程序不仅仅要能够实现所需功能，还要思考如何进一步优化节约程序存储空间，提高程序执行效率，这些可以从数据类型、算法和分支结构等方面具体考虑。当然，将各部分程序组合起来也会出现命名冲突或者地址冲突，这就需要事先和队友沟通好。通过对于插补算法的了解和使用，我对于二维运动控制有了初步认识，本次实习采用的插补方法也比较简单，但我相信在本次实习中获得的经验在今后会帮助更好的学习其他插补方法以及三维运动控制。

总而言之，在小组成员的共同努力下，我们完成了基本功能的设计，并优化了部分功能，较好的完成了实习任务，收获了很多经验。

最后，感谢李勇波老师在实习过程中给予我们的帮助与指导。