

# 《计算机控制技术》课程实验

## 实验一：数字程序控制程序设计

### 【实验目的】

了解数字程序控制原理和数字程序控制方式；

了解插补原理及计算的程序实现、步进驱动控制技术原理及计算的程序实现、伺服驱动控制技术原理及计算的程序实现。

### 【实验设备】

1. 工业控制计算机。
2. 计算机网络

### 【实验原理】

程序流程图：

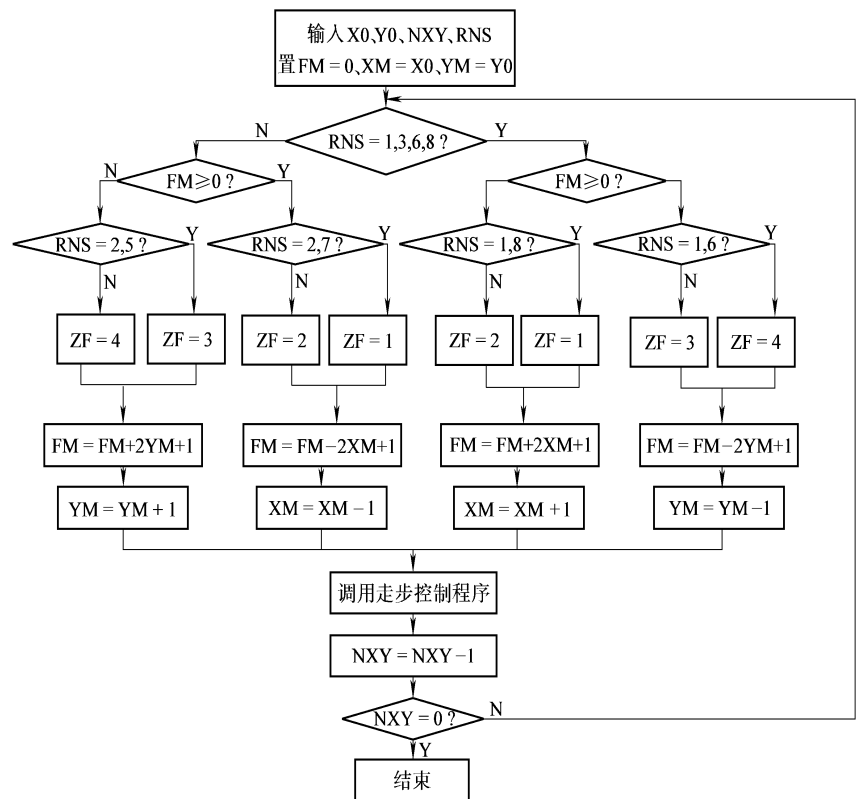
偏差判别

坐标进给

偏差计算

坐标计算

终点判断



### 【实验内容】

1. 四象限圆弧插补计算程序设计
2. 作出走步轨迹图。

### 【实验步骤】

1. 先在 vscode 里面打开基础程序，已经提前配置了 matlab 环境；
2. 显示有些地方有错误，于是按编辑器的提示改正；

- ⚠ 请在(标量)条件语句中使用 && 而不是 & 作为 AND 运算符。 [50, 42]
- ⚠ 请在(标量)条件语句中使用 && 而不是 & 作为 AND 运算符。 [51, 21]
- ⚠ 为了增强可读性, 请在此语句前面使用换行符、分号或逗号。 [51, 35]
- ⚠ 请在(标量)条件语句中使用 && 而不是 & 作为 AND 运算符。 [53, 22]
- ⚠ 为了增强可读性, 请在此语句前面使用换行符、分号或逗号。 [53, 35]
- ⚠ 请在(标量)条件语句中使用 && 而不是 & 作为 AND 运算符。 [55, 21]
- ⚠ 为了增强可读性, 请在此语句前面使用换行符、分号或逗号。 [55, 35]
- ⚠ 请在(标量)条件语句中使用 && 而不是 & 作为 AND 运算符。 [57, 22]
- ⚠ 为了增强可读性, 请在此语句前面使用换行符、分号或逗号。 [57, 35]
- ⚠ 无需使用方括号 []。如果需要, 可使用圆括号进行分组。 [144, 7]

3. 改正后显示没有警告开始运行程序;

4. 按照要求输入对应的起点, 终点坐标, 设置圆弧的半径和方向;

5. 得到实验结果;

```

MATLAB Command Window

要开始, 请键入以下项之一: helpwin、helpdesk 或 demo。
有关产品信息, 请访问 www.mathworks.com。

请输入起点横轴坐标x
x0 = 1
请输入起点纵轴坐标y
y0 = 1
请输入终点横轴坐标x
xe = 2
请输入终点纵轴坐标 y
ye = 2
请输入圆弧半径
R = 5
请选择圆心(1代表靠近原点 2代表远离原点)
1
请选择步走向(1代表顺时针/2代表逆时针)
1
请输入步长
h = 0.02
步长延时
D= 0.1

while ((Xm - Xe)^2 + (Ym - Ye)^2 > h * h / 2 || (step == 0 && Fm == 0))

    if ((Xm - x0) > 0 && (Ym - y0) >= 0); XOY = 1; %判断动点所在象限, 并做标记
    end

    if ((Xm - x0) <= 0 && (Ym - y0) > 0); XOY = 2;
    end

    if ((Xm - x0) < 0 && (Ym - y0) <= 0); XOY = 3;
    end

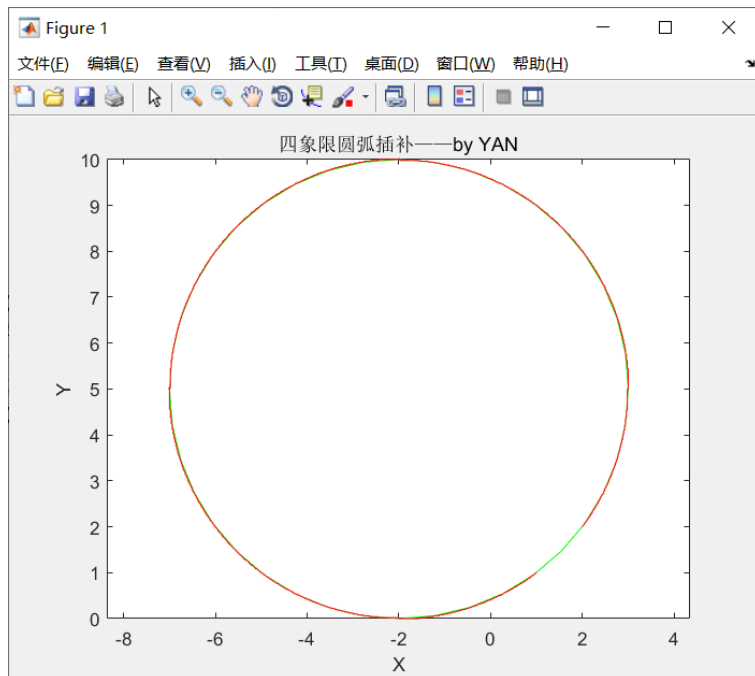
    if ((Xm - x0) >= 0 && (Ym - y0) < 0); XOY = 4;
    end

```

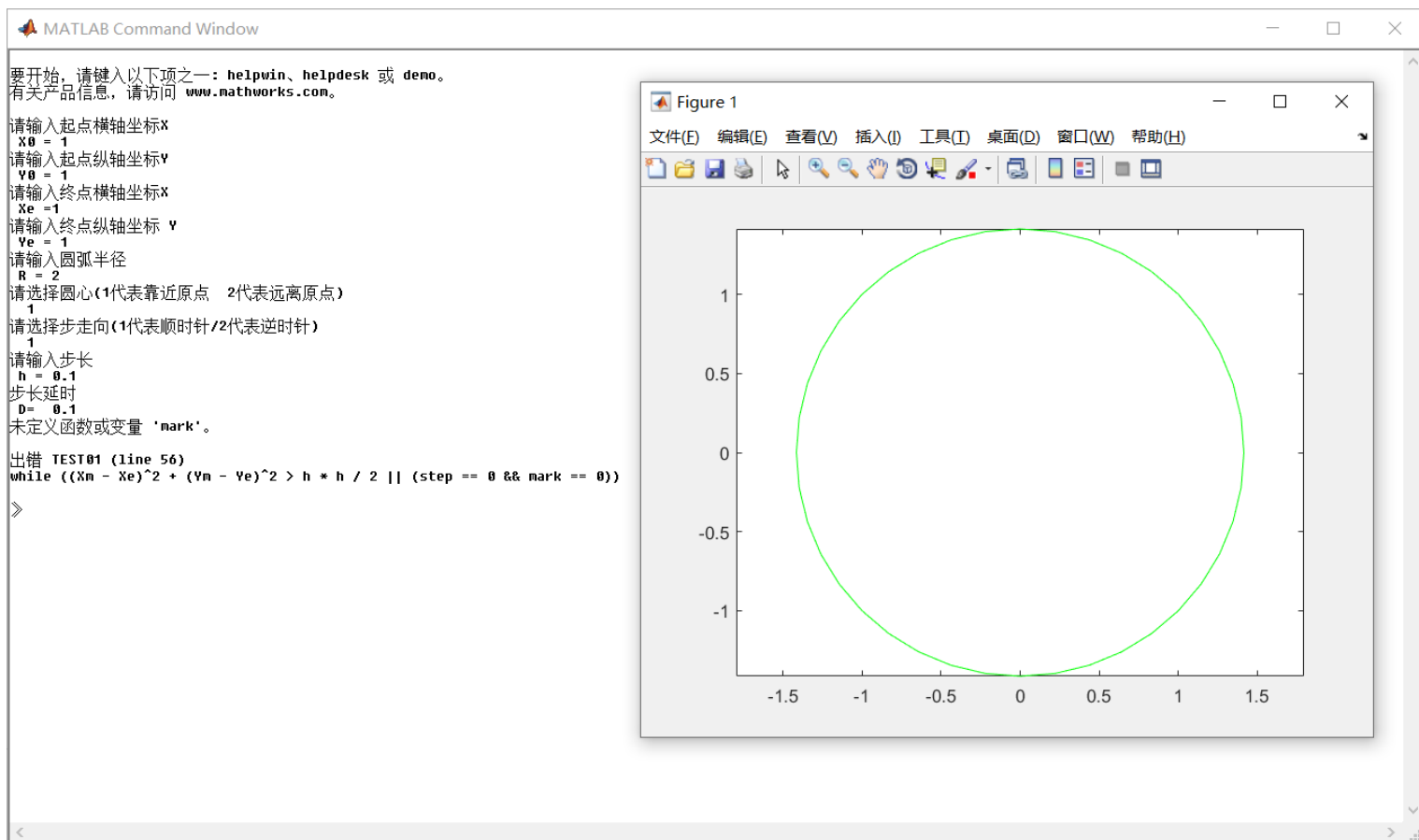
## 【实验报告】

1. 四象限圆弧插补计算过程表；
2. 用 MATLAB 或 C、C++编写四象限圆弧插补计算程序；
3. 作出四象限圆弧插补走步轨迹图。

## 【实验结果】



当起点坐标和终点坐标设置成为一样的时候会报错，同时只画出基准圆；



经检查，将对应的 mark 变量改为 Fm 即可正常运行；

```
while ((Xm - Xe)^2 + (Ym - Ye)^2 > h * h / 2 || (step == 0 && Fm == 0))
```

MATLAB Command Window

要开始，请键入以下项之一：helpwin、helpdesk 或 demo。  
有关产品信息，请访问 [www.mathworks.com](http://www.mathworks.com)。

请输入起点横轴坐标x

X0 = 1

请输入起点纵轴坐标y

Y0 = 1

请输入终点横轴坐标x

Xe = 1

请输入终点纵轴坐标 y

Ye = 1

请输入圆弧半径

R = 1

请选择圆心(1代表靠近原点 2代表远离原点)

1

请选择步走向(1代表顺时针/2代表逆时针)

1

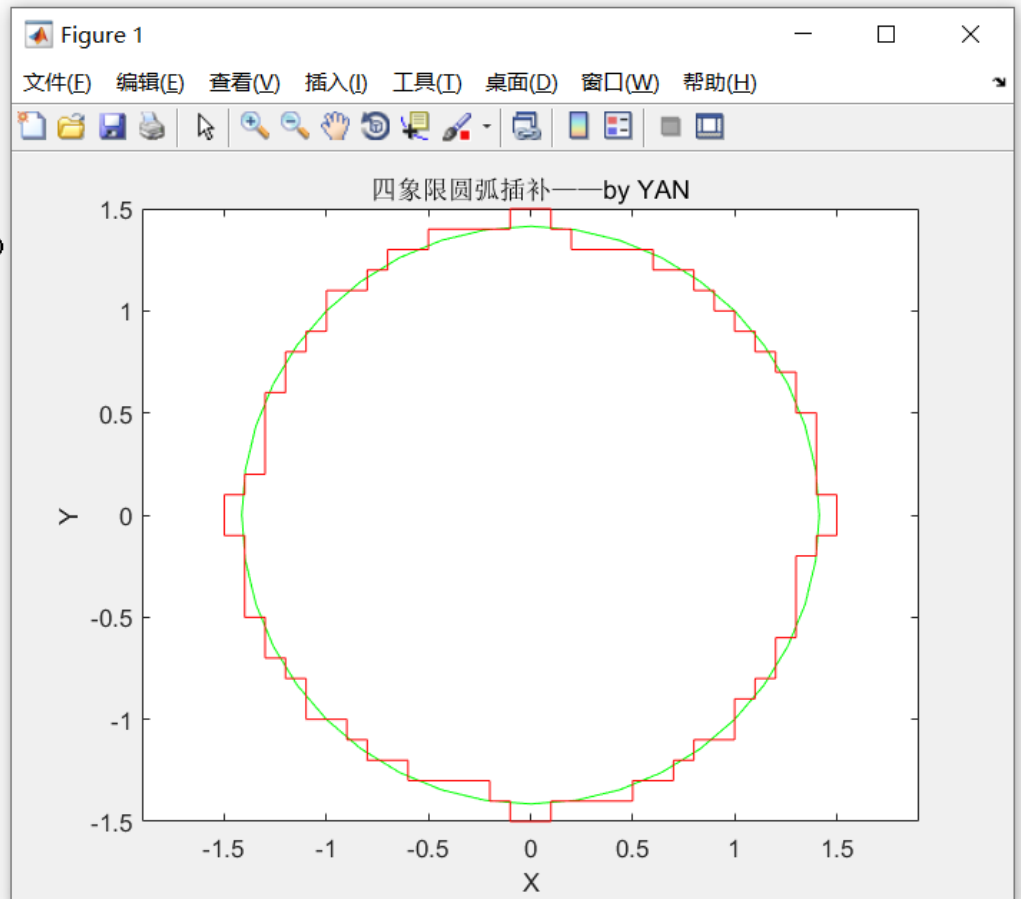
请输入步长

h = 0.1

步长延时

D = 0.1

>>



## 实验程序:

TEST01.m

CCS > TEST01.m > {} while

```
1  X0 = input('请输入起点横轴坐标X\n X0 = ');
2  Y0 = input('请输入起点纵轴坐标Y\n Y0 = ');
3  Xe = input('请输入终点横轴坐标X\n Xe = ');
4  Ye = input('请输入终点纵轴坐标 Y\n Ye = ');
5  R = input('请输入圆弧半径\n R = ');
6  YuanX = input('请选择圆心(1代表靠近原点 2代表远离原点) \n ');
7  ZouX = input('请选择步走向(1代表顺时针/2代表逆时针) \n ');
8  h = input('请输入步长 \n h = ');
9  Delay = input('步长延时 \n D= ');
10 if ((Xe == X0) && (Ye == Y0))
11     x01 = 0; y01 = 0;
12     x02 = 2 * Xe; y02 = 2 * Ye;
13     R = sqrt(Xe^2 + Ye^2);
14 else
15     k1 = (Ye - Y0) / (Xe - X0);
16     k2 = -1 / (k1 + eps); %分母加一个很小的变量，防止出现0/0的情况
17     Xz = (X0 + Xe) / 2; Yz = (Y0 + Ye) / 2; %两点中点坐标
18     L1 = sqrt((X0 - Xe)^2 + (Y0 - Ye)^2) / 2; %两点之间距离的一半
19     L2 = sqrt(R^2 - L1^2);
20     beta = atan(k2);
21     x01 = Xz - L2 * cos(beta); y01 = Yz - L2 * sin(beta); %靠近原点的圆心
22     x02 = Xz + L2 * cos(beta); y02 = Yz + L2 * sin(beta); %远离原点的圆心
23 end
24
25 if (YuanX == 1) %判断圆心位置
26
27     if ((x01^2 + y01^2 - x02^2 - y02^2) <= 0)
28         x0 = x01; y0 = y01;
29     else
30         x0 = x02; y0 = y02;
31     end
32
33 else
34
35     if ((x01^2 + y01^2 - x02^2 - y02^2) <= 0)
36         x0 = x02; y0 = y02;
37     else
38         x0 = x01; y0 = y01;
```

```

37     else
38         x0 = x01; y0 = y01;
39     end
40
41 end
42
43 %画基准圆
44 alpha = 0:pi / 20:2 * pi;
45 xx = R * cos(alpha) + x0;
46 yy = R * sin(alpha) + y0;
47 plot(xx, yy, 'g');
48 hold on;
49 axis equal; %坐标轴的长度单位设为相等
50 Xm = X0;
51 Ym = Y0;
52 %NXY= (abs(Xe-X0)+abs(Ye-Y0))/h;
53 step = 0;
54 Fm = 0;
55
56 while ((Xm - Xe)^2 + (Ym - Ye)^2 > h * h / 2 || (step == 0 && Fm == 0))
57
58     if ((Xm - x0) > 0 && (Ym - y0) >= 0); XOY = 1; %判断动点所在象限，并做标记
59     end
60
61     if ((Xm - x0) <= 0 && (Ym - y0) > 0); XOY = 2;
62     end
63
64     if ((Xm - x0) < 0 && (Ym - y0) <= 0); XOY = 3;
65     end
66
67     if ((Xm - x0) >= 0 && (Ym - y0) < 0); XOY = 4;
68     end
69
70     switch XOY
71     case 1
72
73         if (ZouX == 1)
74
75             if (Fm >= 0)
76                 x1 = [Xm, Xm];
77                 y1 = [Ym, Ym - h];
78             else
79                 x1 = [Xm, Xm + h];
80                 y1 = [Ym, Ym];
81             end
82
83         else
84
85             if (Fm <= 0)
86                 x1 = [Xm, Xm];
87                 y1 = [Ym, Ym + h];
88             else
89                 x1 = [Xm, Xm - h];
90                 y1 = [Ym, Ym];
91             end
92
93         end
94

```

```

94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183

end
case 2
    if (ZouX == 1)
        if (Fm >= 0)
            x1 = [Xm, Xm + h];
            y1 = [Ym, Ym];
        else
            x1 = [Xm, Xm];
            y1 = [Ym, Ym + h];
        end
    else
        if (Fm > 0)
            x1 = [Xm, Xm];
            y1 = [Ym, Ym - h];
        else
            x1 = [Xm, Xm - h];
            y1 = [Ym, Ym];
        end
    end
end
case 3
    if (ZouX == 1)
        if (Fm >= 0)
            x1 = [Xm, Xm];
            y1 = [Ym, Ym + h];
        else
            x1 = [Xm, Xm - h];
            y1 = [Ym, Ym];
        end
    else
        if (Fm > 0)
            x1 = [Xm, Xm + h];
            y1 = [Ym, Ym];
        else
            x1 = [Xm, Xm];
            y1 = [Ym, Ym - h];
        end
    end
end
case 4
    if (ZouX == 1)
        if (Fm >= 0)
            x1 = [Xm, Xm - h];
            y1 = [Ym, Ym];
        else
            x1 = [Xm, Xm];
            y1 = [Ym, Ym - h];
        end
    else
        if (Fm > 0)
            x1 = [Xm, Xm];
            y1 = [Ym, Ym + h];
        else
            x1 = [Xm, Xm + h];
            y1 = [Ym, Ym];
        end
    end
end

step = step + 1;
plot(x1, y1, 'r'); %由此点和前一点坐标组成的2个向量画直线
Xm = x1(2); %保存此点坐标供下次作图和比较时使用
Ym = y1(2);
Fm = (Xm - x0)^2 + (Ym - y0)^2 - R^2;
hold on;
%text((x1(1)+x1(2))/2, (y1(1)+y1(2))/2, [num2str(step)]) %给每一步标号
pause(Delay); %延时程序形参为每走一步所用时间
end

xlabel('X')
ylabel('Y')
title('四象限圆弧插补—by YAN');
hold off;

```