# 计算机图形学期中实验报告

1. **实验内容**

编程实现在控制台绘制出各种图形，可以进行简单的人机交互（即人为选择绘制图形）。熟悉使用各种二维图形的生成算法，包括三种直线生成算法，六种二次曲线的算法，自由曲线算法以及简单的喷枪区域效果图（中间像素点密集，向外分散的圆形区域）。

1. **实验环境**

CodeBlocks ：：ege 环境下，C语言编码

1. **实验源代码**

#include "graphics.h"

#include "stdio.h"

#include "math.h"

void p1(){

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\* 二维图形的生成 \*\n");

printf("\* 1.直线 \*\n");

printf("\* 2.二次曲线 \*\n");

printf("\* 3.自由曲线 \*\n");

printf("\* 4.简单喷枪效果 \*\n");

printf("\* 5.退出 \*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

void p2(){

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\* 直线生成算法 \*\n");

printf("\* 1.逐点比较法 \*\n");

printf("\* 2.数值微分DDA法\*\n");

printf("\* 3.bresenham算法\*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

void p3(){

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\* 二次曲线生成算法\*\n");

printf("\* 1.中点画圆法 \*\n");

printf("\* 2.DDA法画圆弧 \*\n");

printf("\* 3.bresenham画圆\*\n");

printf("\* 4.椭圆轨变 \*\n");

printf("\* 5.DDA截取椭圆弧\*\n");

printf("\* 6.二次曲线拟合 \*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

void p4(){

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\* 自由曲线生成算法 \*\n");

printf("\* 1.抛物线参数样条曲线\*\n");

printf("\* 2.贝塞尔(Bezier)曲线\*\n");

printf("\* 3.三、Bezier曲线 \*\n");

printf("\* 4.B样条曲线 \*\n");

printf("\* 5.三、B样条曲线 \*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

void zhudian(int x1,int y1,int x2,int y2) {//zhudianfa

int x,y,i,n; int f;

n=(x2-x1)+(y2-y1);

x=x1; y=y1;

f=y\*x2-y2+x;

for (i=1; i<=n; i++){

putpixel(x,y,RED,NULL);

if (f>=0) {

x=x+1;

y=y;

}

else{

x=x;

y=y+1;

}

f=y\*x2-y2\*x;

}

}

#define ROUND(a) ((int)(a+0.5))

void DDA(int x1, int y1, int x2, int y2){//数值微分法（DDA法）

int i; float x,y,k;

k=(y2-y1)/(x2-x1);

x=x1; y=y1;

if(k<=1)

for (i=x1 ; i<=x2 ; i++){

putpixel(ROUND(x),ROUND(y),RED,NULL);

x=x+1;

y=y+k;

}

else

for (i=y1;i<=y2;i++){

putpixel(ROUND(x), ROUND(y),RED,NULL);

x=x+1/k;

y=y+1;

}

}

void bresenham(int x1, int y1, int x2, int y2)//bresenham

{ int dx, dy; int d;

int inc1, inc2, x,y, xend;

dx=abs(x1-x2); dy=abs(y1-y2); d=2\*dy-dx;

inc1=2\*dy; inc2=2\*(dy-dx);

if(x1>x2) //定义起点和终点/

{ x=x2; y=y2; xend=x1;}

else{

x=x1; y=y1; xend=x2;}

putpixel(x,y,RED,NULL);

while (x<xend) {

x++;

if (d<0) {d=d+inc1;}

else {y++; d=d+inc2;}

putpixel(x,y,RED,NULL);

}

}

void midpointc(int xc, int yc, int r){

int x=0; int y=r; int p=1-r;

void circleplotpoints(int q, int w, int e, int r);

circleplotpoints(xc, yc, x,y);

while(x<y) { x++;

if ( p<0)

p+=2\*x+1;

else { y--;

p+=2\*(x-y)+1;

}

circleplotpoints(xc, yc, x,y);

}

}

void circleplotpoints (int xc, int yc, int x, int y){

putpixel(xc+x, yc+y,RED,NULL);

putpixel(xc-x, yc+y,RED,NULL);

putpixel(xc+x, yc-y,RED,NULL);

putpixel(xc-x, yc-y,RED,NULL);

putpixel(xc+y, yc+x,RED,NULL);

putpixel(xc-y, yc+x,RED,NULL);

putpixel(xc+y, yc-x,RED,NULL);

putpixel(xc-y, yc-x,RED,NULL);

}

void DDAyh(int xc, int yc, double r, double ts, double te){

double rad, tsl ,tel, deg, dte,ta, ct,st; int x,y,n,i;

rad=0.0174533; tsl=ts\*rad; tel=te\*rad;

if(r<5.08) deg=0.015;

else if(r<7.62)

deg=0.06;

else if(r<25.4)

deg=0.075;

else deg=0.15;

dte=deg\*25.4/r;

if(tel<tsl) tel=tel+6.28319;

n=(int)((tel-tsl)/dte+0.5);

if(n==0) n=(int)(6.28319/dte+0.5);

ta=tsl;

x=xc+r\*cos(tsl); y=yc+r\*sin(tsl);

moveto(x,y);

for(i=1; i<=n; i++) {

ta=ta+dte; ct=cos(ta); st=sin(ta);

x=xc+r\*ct; y=yc+r\*st;

lineto(x,y);

}

x=xc+r\*cos(tel); y=yc+r\*sin(tel);

lineto(x,y);

}

void drawpot(int x0,int y0,int x,int y){

putpixel(x0+x,y0+y,RED,NULL);

putpixel(x0+y,y0+x,RED,NULL);

putpixel(x0+y,y0-x,RED,NULL);

putpixel(x0+x,y0-y,RED,NULL);

putpixel(x0-x,y0-y,RED,NULL);

putpixel(x0-y,y0-x,RED,NULL);

putpixel(x0-y,y0+x,RED,NULL);

putpixel(x0-x,y0+y,RED,NULL);

}

void bresenhamc(int x0,int y0,int r){

int x=0, y=r, d=3-2\*r;

while(x<y){

drawpot(x0,y0,x,y);

if(d<0) d=d+4\*x+6;

else{

d=d+4\*(x-y)+10; y--;

}

x++;

}

}

void tyguibian ( int x0,int y0,int a,int b,int dt){

int x,y,n,i; float t1,t=0.0;

t1=dt\*0.0174533; n=360/dt;

moveto(x0+1,y0);

for(i=1;i<n;i++){

t=t+t1;

x=x0+a\*cos(t); y=y0+b\*sin(t);

lineto(x,y);

}

lineto(x0+a,y0);

}

void DDAjietyh(int xc, int yc, int a, int b, double alp, double ts, double te){

double rad, tsl, tel, alpl, deg, dte, r, ta, a1,a2,b1,b2;

int x,y,n,i;

rad=0.0174533; tsl=ts\*rad; tel=te\*rad; alpl=alp\*rad;

a1=a\*cos(tsl); b1=cos(alpl); a2=b\*sin(tsl); b2=sin(alpl);

r=(a>b)?a:b;

if (r<5.08) deg=0.015;

else if (r>7.62)

deg=0.06;

else if (r<25.4)

deg=0.075;

else deg=0.15;

dte=deg\*25.4/r;

if (tel<tsl) tel+=6.28319;

n=(int)(6.28319/dte+0.5);

ta=tsl;

x=xc+a1\*b1-a2\*b2; y=yc+a1\*b2+a2\*b1;

moveto(x,y);

for (i=1;i<=n;i++){

ta+=dte;

a1=a\*cos(ta); a2=b\*sin(ta);

x=xc+a1\*b1-a2\*b2; y=yc+a1\*b2+a2\*b1;

lineto (x,y);

}

a1=a\*cos(tel); a2=b\*sin(tel);

x=xc+a1\*b1-a2\*b2; y=yc+a1\*b2+a2\*b1;

lineto(x,y);

}

void ecqxnihe(int xs, int ys, int xm, int ym, int xe, int ye){

double d, d1, ax, ay, bx,by; int n,i;

ax=(xe-2\*xm+xs)\*2.0; ay=(ye-2\*ym+ys)\*2.0;

bx=(xe-xs-ax); by=(ye-ys-ay);

n=sqrt(ax\*ax+ay\*ay)/4; n=sqrt(n\*100);

moveto (xs,ys);

d=1.0/n; d1=d;

for (i=0; i<=n;i++) {

lineto ((int)( ax\*d1\*d1+bx\*d1+xs), (int)(ay\*d1\*d1+by\*d1+ys));

d1=d1+d;

}

lineto(xe,ye);

}

void parspl(int p[][2],int n,int k,int e){

int x,y,i,j,m; float t1,t2,t3,t,a,b,c,d;

if(e==1){

m=n;

p[0][0]=p[1][0]; p[0][1]=p[1][1];

p[n+1][0]=p[n][0];p[n+1][1]=p[n][1];

}

else{

m=n+1;

p[0][0]=p[n][0]; p[0][1]=p[n][1];

p[m][0]=p[1][0]; p[m][1]=p[1][1];

p[m+1][0]=p[2][0];p[m+1][1]=p[2][1];

}

t=0.5/k;

moveto(p[1][0],p[1][1]);

for(i=0;i<m-1;i++){

for(j=1;j<k;j++){

t1=j\*t; t2=t1\*t1; t3=t2\*t1;

a=4.0\*t2-t1-4.0\*t3; b=1.0-10.0\*t2+12.0\*t3;

c=t1+8.0\*t2-12.0\*t3; d=4.0\*t3-2.0\*t2;

x=a\*p[i][0]+b\*p[i+1][0]+c\*p[i+2][0]+d\*p[i+3][0];

y=a\*p[i][1]+b\*p[i+1][1]+c\*p[i+2][1]+d\*p[i+3][1];

lineto(x,y);

}

lineto(p[i+2][0],p[i+2][1]);

}

}

void marking(int a[][2],int n){

int i;

setfillstyle(1,WHITE);

for(i=1;i<=n;i++){

circle(a[i][0],a[i][1],2);

floodfill(a[i][0],a[i][1],RED);

}

}

void bezz(int x[],int y[],int n){

float c[10],b[10],p[10],q[10],xe,ye;

int i,j,k; float t;

moveto(x[0],y[0]);

for(i=0;i<=n;i++) c[i]=1;

for(i=2;i<=n;i++){

for(j=1;j<=i-1;j++) b[j]=c[j-1]+c[j];

for(j=1;j<=i-1;j++) c[j]=b[j];

}

for(i=0;i<=n\*4;i++){

t=(float)i/(n\*4);

p[0]=1;q[0]=1;

for(j=1;j<=n;j++){

p[j]=p[j-1]\*t;

q[j]=q[j-1]\*(1-t);

}

for(j=0;j<=n;j++) b[j]=p[j]\*q[n-j];

xe=0;ye=0;

for(k=0;k<=n;k++){

xe=xe+c[k]\*b[k]\*x[k];

ye=ye+c[k]\*b[k]\*y[k];

}

lineto(xe,ye);

}

}

void mark(int x[],int y[],int m){

int k;

moveto(x[0],y[0]);

for(k=0;k<m;k++){

if(k<m-1)

line(x[k],y[k],x[k+1],y[k+1]);

circle(x[k],y[k],RED);

}

}

void b(int u[],int v[],int n1){

int i,j,k,num,xe,ye;

float t,b[4][10],x[10],y[10];

num=15;

for(i=0;i<10;i++){

t=(float)i/num;

b[0][i]=(1-t)\*(1-t)\*(1-t)/6;

b[1][i]=(t-2)\*t\*t/2+2.0/3;

b[2][i]=(1+t-t\*t)\*t/2.0+1.0/6;

b[3][i]=t\*t\*t/6.0;

}

for(i=1;i<=n1;i++){

x[i]=u[i-1]; y[i]=v[i-1];

}

x[0]=2\*x[1]-x[2]; y[0]=2\*y[1]-y[2];

x[n1+1]=2\*x[n1]-x[n1-1];

y[n1+1]=2\*y[n1]-y[n1-1];

putpixel(x[1],y[1],4);

moveto(x[1],y[1]);

for(i=0;i<=n1-1;i++){

for(j=0;j<10;j++){

xe=0;ye=0;

for(k=0;k<=3;k++){

xe=xe+b[k][j]\*x[i+k];

ye=ye+b[k][j]\*y[i+k];

}

lineto(xe,ye);

}

moveto(xe,ye);

}

}

int main(){

while(true){

p1();

int m1,m2;

printf("请选择一级目录 : ");

scanf("%d",&m1);

if (m1 == 1){

p2();

int x1,y1,x2,y2;

printf("请选择直线目录: ");

scanf("%d",&m2);

printf("输入直线上两个点的坐标 :");

scanf("%d,%d,%d,%d",&x1,&y1,&x2,&y2);

switch(m2){

case 1: initgraph(640,480); setbkcolor(WHITE);

zhudian(x1,y1,x2,y2); break;

case 2: initgraph(640,480); setbkcolor(WHITE);

DDA(x1,y1,x2,y2); break;

case 3: initgraph(640,480); setbkcolor(WHITE);

bresenham(x1,y1,x2,y2); break;

}

}

if (m1 == 2){

p3();

int xc,yc,m3,a,b,dt,r1;

int xs,ys,xm,ym,xe,ye;

double r,ts,te; double alp;

printf("选择二次曲线目录 : ");

scanf("%d",&m3);

switch(m3){

case 1: printf("请输入圆心坐标和半径 :");

scanf("%d,%d,%lf",&xc,&yc,&r);

initgraph(640,480); setbkcolor(WHITE);

midpointc(xc,yc,r); break;

case 2: printf("输入其所在圆的圆心坐标及半径，以水平轴为准的开始角度及结束角度:\n");

scanf("%d,%d,%lf,%lf,%lf",&xc,&yc,&r,&ts,&te);

initgraph(640,480); setbkcolor(WHITE); setcolor(RED);

DDAyh(xc,yc,r,ts,te); break;

case 3: printf("请输入圆心坐标和半径 :");

scanf("%d,%d,%d",&xc,&yc,&r1);

initgraph(640,480); setbkcolor(WHITE);

bresenhamc(xc,yc,r1); break;

case 4: printf("输入椭圆中心点坐标，长短轴及轨变幅度 :");

scanf("%d,%d,%d,%d,%d",&xc,&yc,&a,&b,&dt);

setbkcolor(WHITE); setcolor(RED);

for(int i=0;i<=a;i=i+dt)

tyguibian (xc,yc,a-i,i,dt);

break;

case 5: printf("输入要截取的一段椭圆弧的数据 :");

scanf("%d,%d,%d,%d,%lf,%lf,%lf",&xc,&yc,&a,&b,&alp,&ts,&te);

initgraph(640,480); setbkcolor(9); setcolor(RED);

DDAjietyh(xc,yc,a,b,alp,ts,te); break;

case 6: printf("请输入三个点的坐标 :");

scanf("%d,%d,%d,%d,%d,%d",&xs,&ys,&xm,&ym,&xe,&ye);

initgraph(640,480); setbkcolor(9); setcolor(RED);

ecqxnihe(xs,ys,xm,ym,xe,ye); break;

}

}

if (m1 == 3){

p4();

int m4;

int i,n,c,p[50][2];

int m,j; int x[10],y[10];

int xx[4]={100,200,400,500},yy[4]={100,300,200,100};

float t,xs,ys,xe,ye; float b03,b13,b23,b33;

int u[5]={120,180,240,300,360},v[5]={220,50,200,260,280},n1=5;

int n2;

int x1[4]={100,200,400,500},y1[4]={100,300,200,100};

float nn0,nn1,nn2,nn3; float u1;

printf("请选择自由曲线目录 : ");

scanf("%d",&m4);

switch(m4){

case 1:

printf("请输入点的个数:");

scanf("%d",&n);

for(i=1;i<=n;i++)

scanf("%d,%d",&p[i][0],&p[i][1]);

printf("Input:FREE----1; CLOSE----2;\n");

scanf("%d",&c);

initgraph(640,480); setbkcolor(9); setcolor(RED);

marking(p,n);

parspl(p,n,10,c); break;

case 2:

printf("请输入点的个数:");

scanf("%d",&m);

printf("请输入各个点坐标:\n");

for(j=0;j<m;j++)

scanf("%d,%d",&x[j],&y[j]);

initgraph(640,480); setbkcolor(9); setcolor(RED);;

mark(x,y,m);

n=4;

setcolor(4);

bezz(x,y,m-1);

break;

case 3:

printf("请输入各个点坐标:\n");

for(i=0;i<4;i++)

scanf("%d,%d",&xx[i],&yy[i]);

xs=xx[0]; ys=yy[0];

initgraph(640,480); setbkcolor(9); setcolor(RED);

moveto(xs,ys);

for(i=0;i<4;i++){

lineto(xx[i],yy[i]);

}

for(i=0;i<4;i++)

circle(xx[i],yy[i],5);

moveto(xs,ys);

for(t=0.0 ;t<=1.0001;t=t+0.01){

b03=(1-t)\*(1-t)\*(1-t);

b13= 3\*t\*(1-t)\* (1-t);

b23= 3\*t\*t\*(1-t);

b33= t\*t\*t;

xe=b03\*xx[0]+b13\*xx[1]+b23\*xx[2]+b33\*xx[3];

ye=b03\*yy[0]+b13\*yy[1]+b23\*yy[2]+b33\*yy[3];

lineto(xe,ye);

xs=xe;ys=ye;

}

break;

case 4: initgraph(640,480); setbkcolor(9); setcolor(RED);

b(u,v,n1); break;

case 5: n2 = 3;

printf("请输入各个点坐标:\n");

for(i=0;i<4;i++)

scanf("%d,%d",&x1[i],&y1[i]);

xs= (x1[0] + x1[2]) \* 0.5 / 3 + 2 \* x1[1] / 3 ;

ys= (y1[0] + y1[2]) \* 0.5 / 3 + 2 \* y1[1] / 3;

initgraph(640,480); setbkcolor(9); setcolor(RED);

moveto(x1[0],y1[0]);

for(i=0;i<4;i++){

lineto(x1[i],y1[i]);

}

for(i=0;i<4;i++)

circle(x1[i],y1[i],5);

for(i=0 ;i<=n2-3;i++)

for(u1=0.01;u1<=1.00001;u1=u1+0.01){

nn0 = (-u1\*u1\*u1 + 3\*u1\*u1 - 3\*u1 + 1) / 6;

nn1 = (3\*u1\*u1\*u1 - 6\*u1\* u1 + 4) / 6;

nn2 = (-3\*u1\*u1\*u1 + 3\*u1\*u1 + 3\*u1 + 1) / 6;

nn3 = u1\*u1\*u1/6;

xe = x1[i] \*nn0 + x1[i + 1] \*nn1 + x1[i + 2] \*nn2 + x1[i + 3] \*nn3;

ye = y1[i] \*nn0 + y1[i + 1] \*nn1 + y1[i + 2] \*nn2 + y1[i + 3] \*nn3 ;

moveto(xs,ys);

lineto(xe, ye);

xs =xe; ys = ye;

}

break;

}

}

if (m1 == 4){

int x0,y0,r;

printf("请输入喷枪区域的大小 :");

scanf("%d,%d,%d",&x0,&y0,&r);

initgraph(640,480); setbkcolor(WHITE);

for (int i=x0-r;i<x0+r;i++){

for (int j=y0-r;j<y0+r;j++){

int t = (x0-i)\*(x0-i)+(y0-j)\*(y0-j);

if (t<r\*r)

if (rand()%(r\*r)>=t){

putpixel(i,j,RED);

}

}

}

continue;

}

if (m1 == 5){

break;

}

getch();

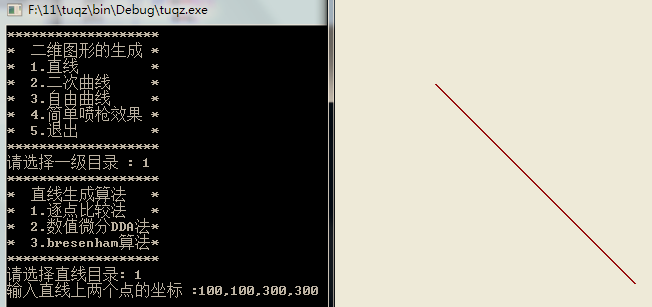
closegraph();

}

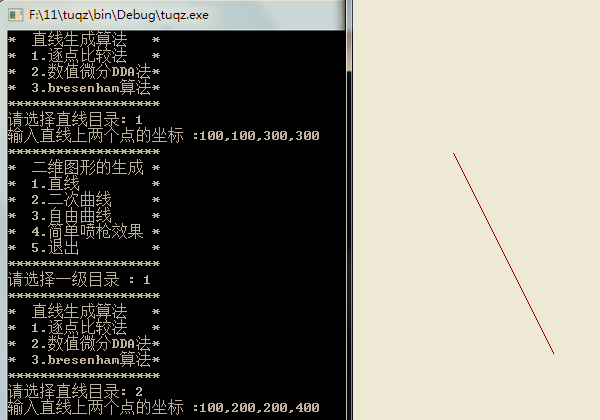
return 0;

}

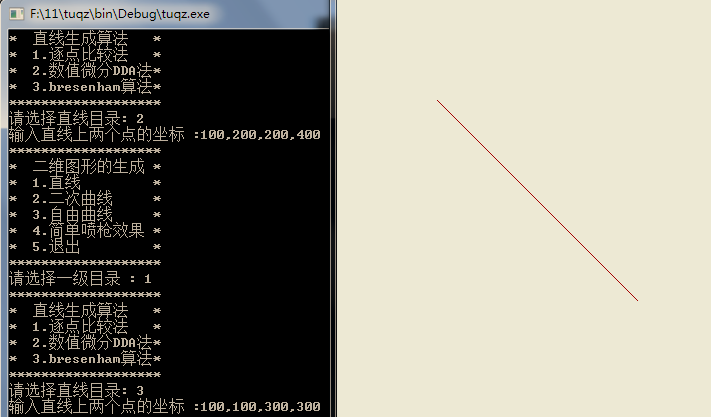
1. **实验结果截图**
2. 三种直线生成算法
3. 逐点法



（2）数值微分DDA法

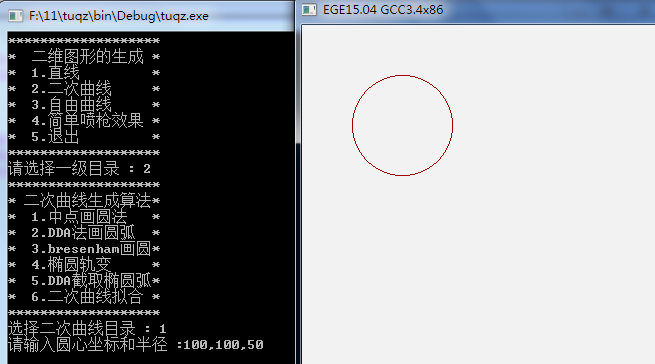


1. Bresenham算法

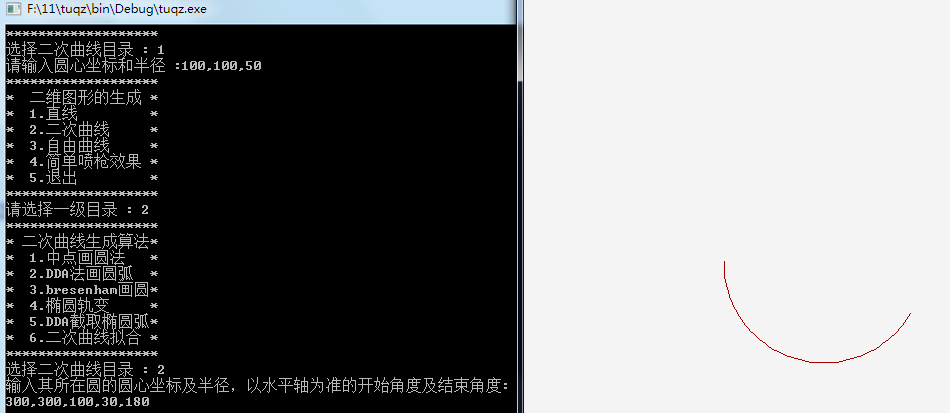


2.二次曲线

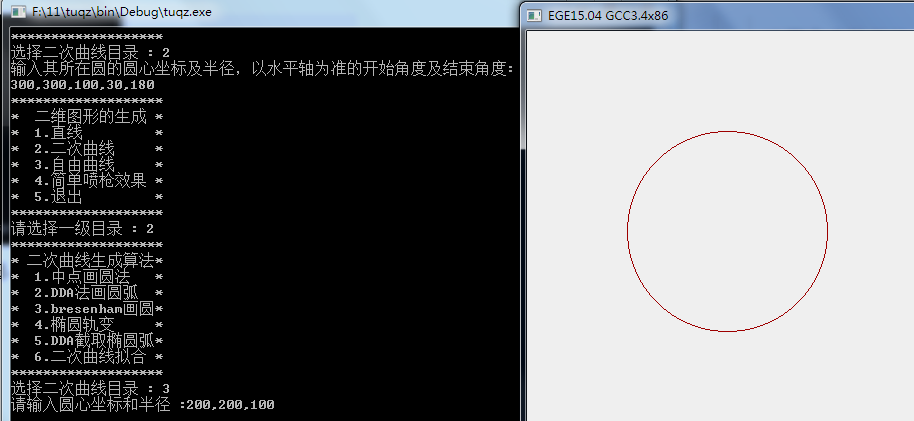
（1）中心点画圆



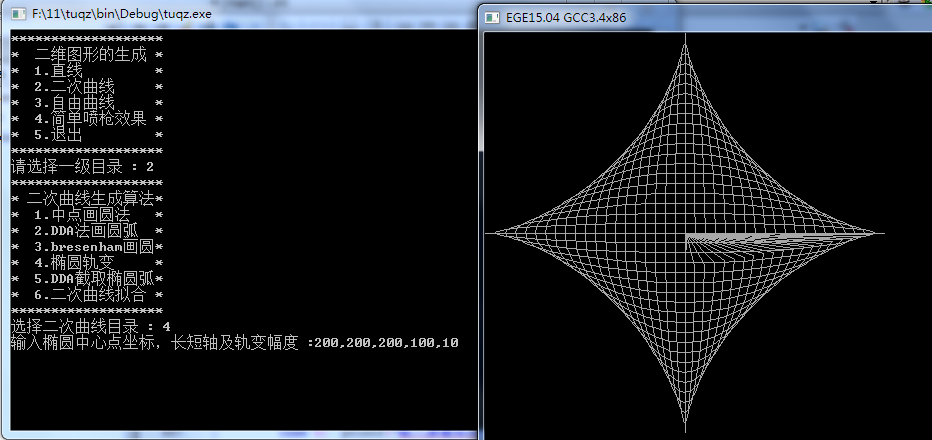
1. DDA画圆弧



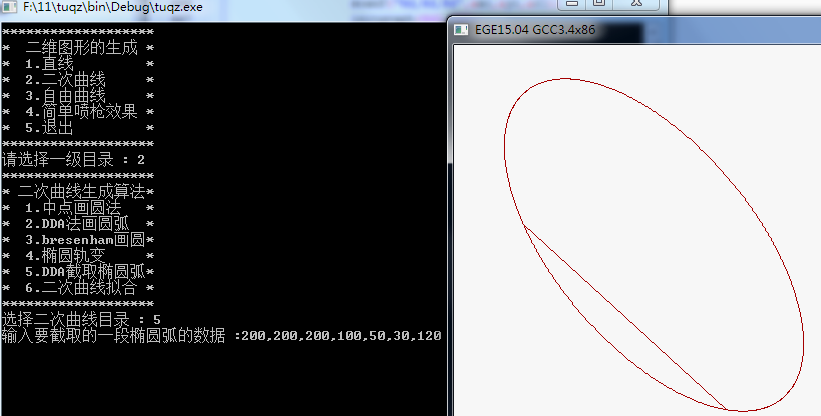
（3）Bresenham画圆



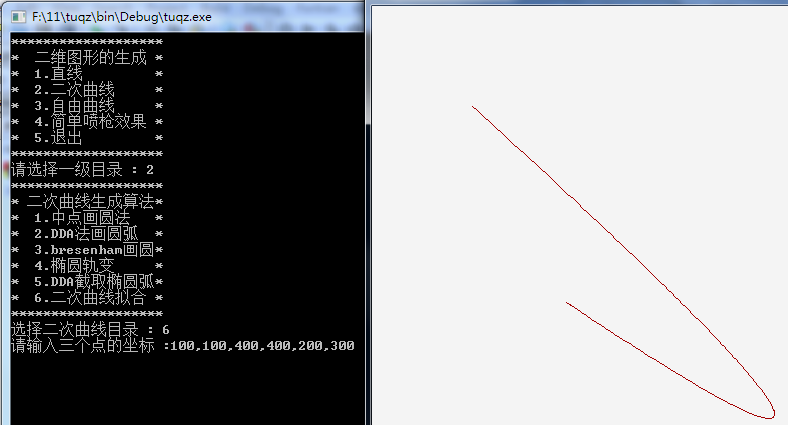
（4）椭圆的轨变



（5）DDA截取一段椭圆圆弧

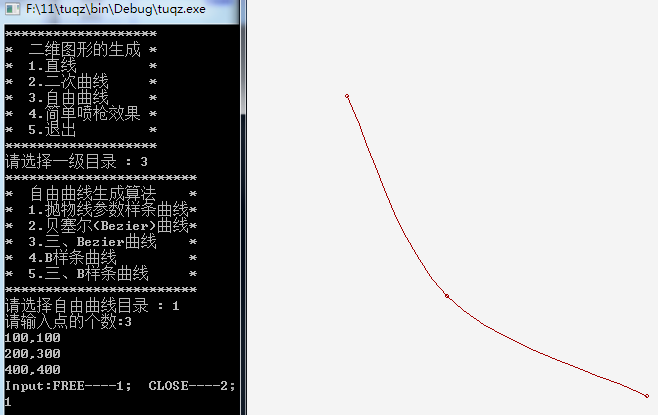


（6）二次曲线参数拟合

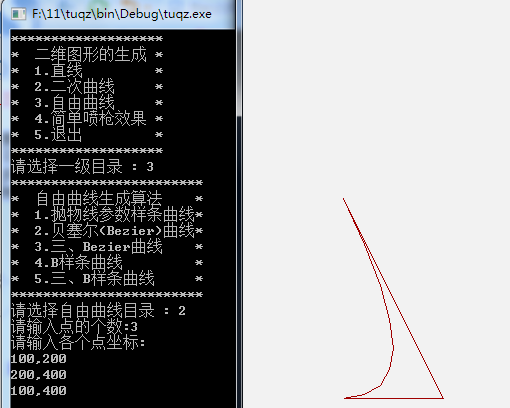


3.自由曲线

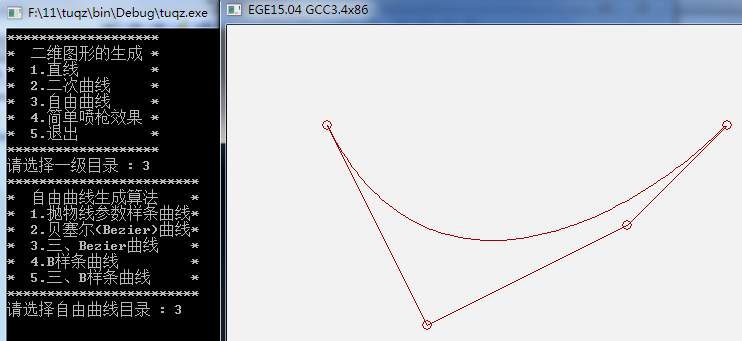
（1）抛物线参数样条曲线



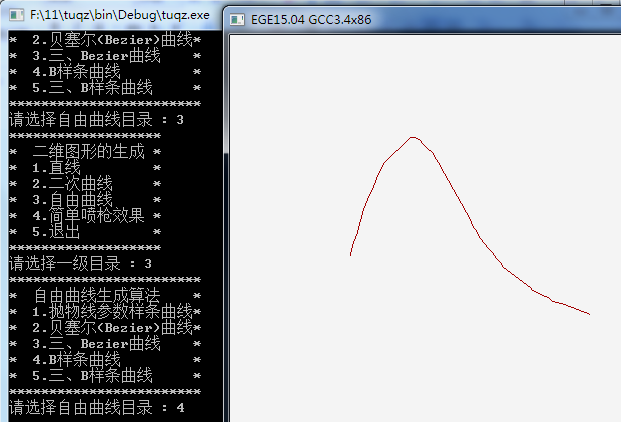
（2）贝塞尔(Bezier)曲线



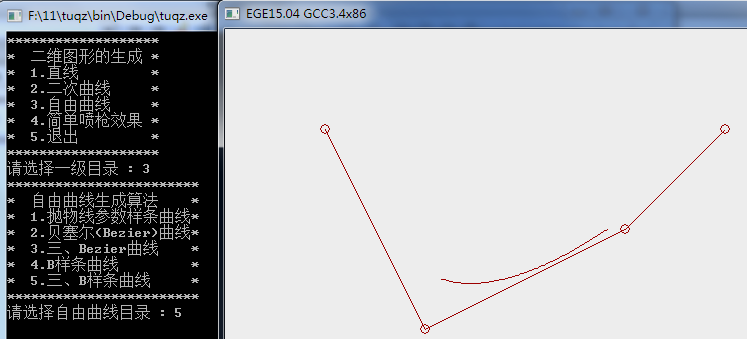
（3）三、Bezier曲线



（4）B样条曲线



（5）三、B样条曲线



4.简单的喷枪区域效果图（中间像素点密集，外围像素点分散）

