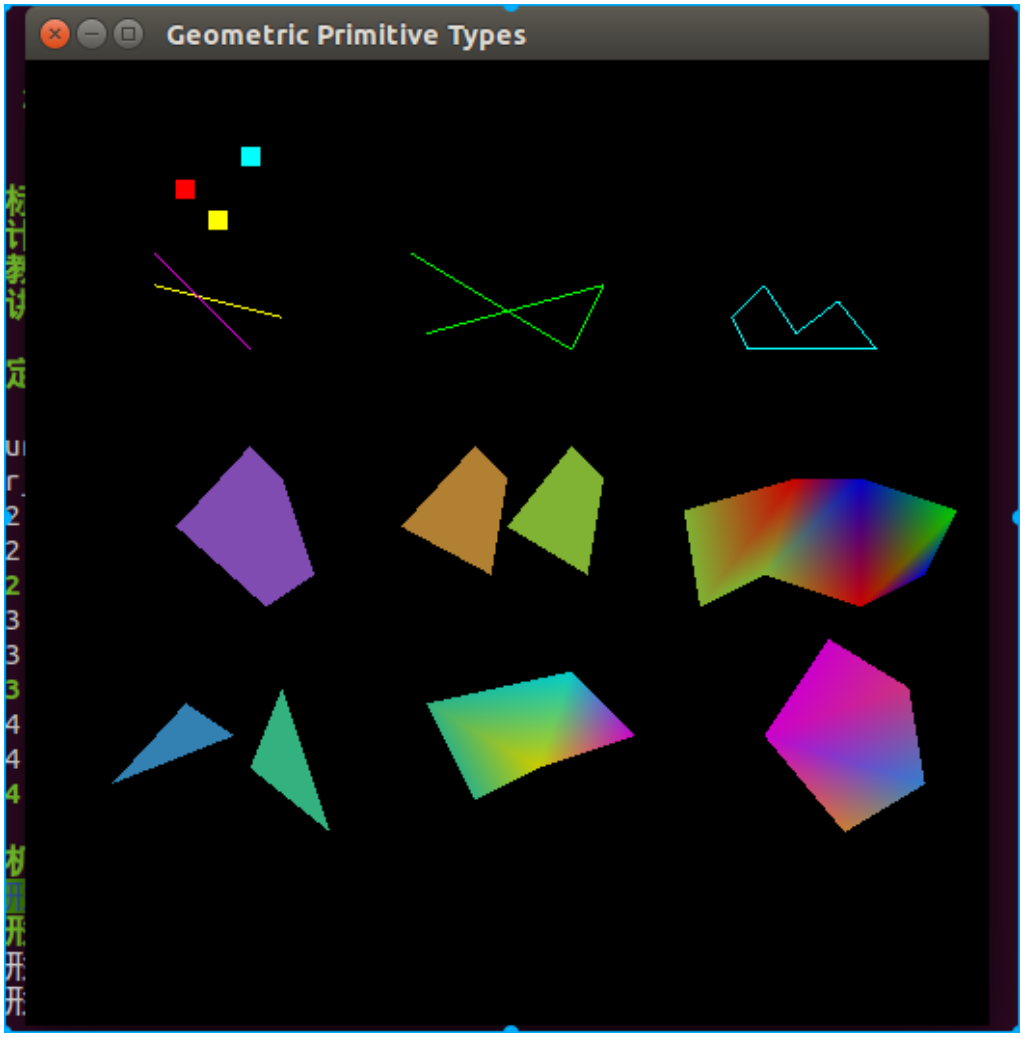
1. 在屏幕上绘制几何图元，自定义坐标和颜色

1.1. 实验目的

1. 理解glut程序框架，能使用opengl绘制基本几何元件
2. 实现直线段的中点算法，加深对算法的理解
3. 实现生成圆的中点算法，加深对算法的理解

1.2. 实验结果



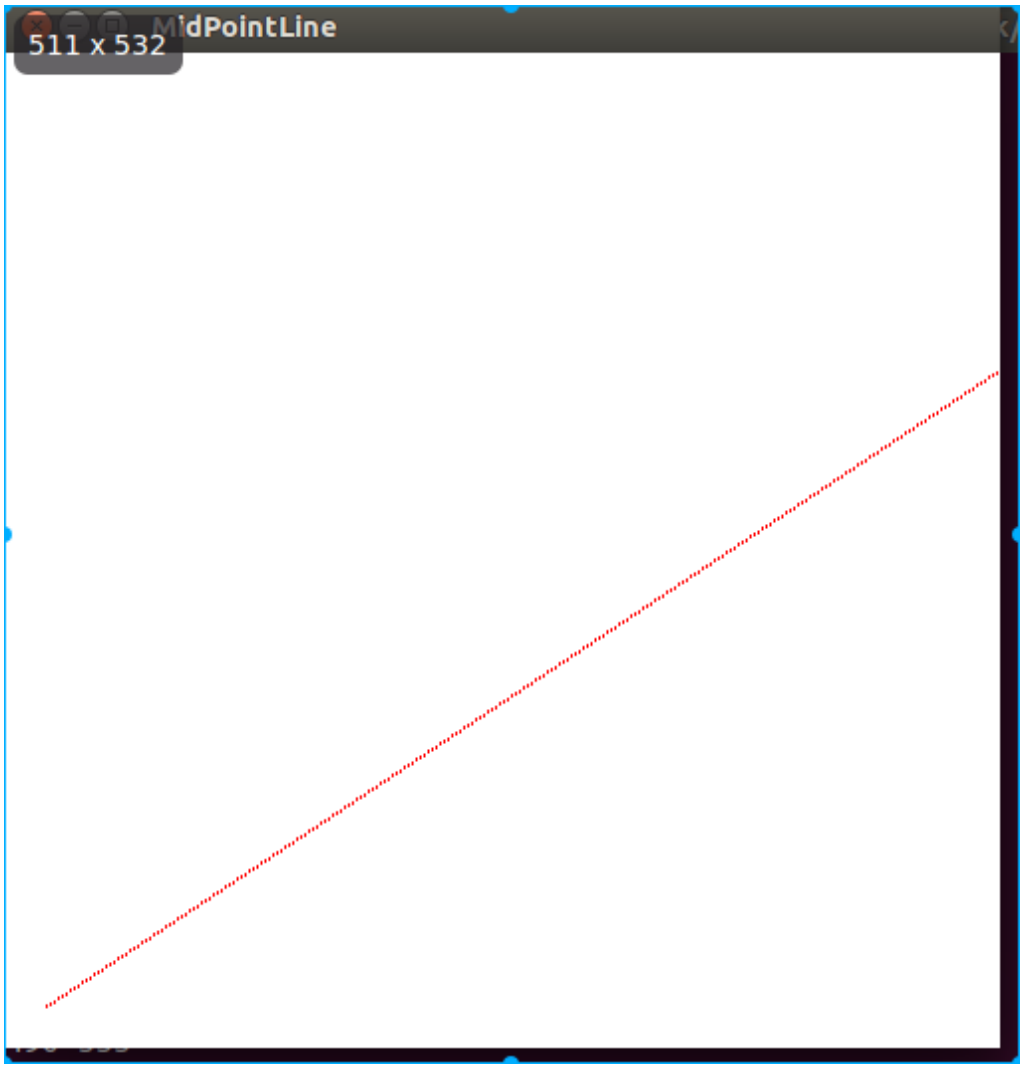
2. 算法模拟题（可二选一实现）

1. 采用中点线算法在屏幕上画一条直线。
2. 采用中点圆算法在屏幕上画一个圆。

2.1. 实验目的

实现采用中点线算法在屏幕上画一条直线。

2.2. 实验结果



3. 心得体会

通过本次上机实验，学会了OpenGL中关于绘制几何图元的各个函数的应用，同时通过自己亲手实现了中点线算法，加深了对其原理的理解，和课堂上学到的知识相互印证，巩固了学习成果。

4. 实验源码

4.1. 题目一

```
#coding:utf8
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *


def init():
    glClearColor(0.0,0.0,0.0,0.0);#
    gluOrtho2D(-15.0,15.0,-15.0,15.0)
```

```
def RenderScene():
    glClear(GL_COLOR_BUFFER_BIT);#清除屏幕
    glColor3f(1.0,1.0,0.0);#设置颜色
    DrawMyObjects()
    glFlush();

def DrawMyObjects():
    #画点
        glPointSize(10);
    glBegin(GL_POINTS);
    glColor3f(1.0,0.0,0.0);
    glVertex2f(-10.0,11.0);
    glColor3f(1.0,1.0,0.0);
    glVertex2f(-9.0,10.0);
    glColor3f(0.0,1.0,1.0);
    glVertex2f(-8.0,12.0);
    glEnd();
#        glPointSize(10);
    #画线段
    glBegin(GL_LINES);
    glColor3f(1.0,1.0,0.0);

    glVertex2f(-11.0,8.0);
    glVertex2f(-7.0,7.0);
    glColor3f(1.0,0.0,1.0);
    glVertex2f(-11.0,9.0);
    glVertex2f(-8.0,6.0);
    glEnd();
    #画开折线
    glBegin(GL_LINE_STRIP);
    glColor3f(0.0,1.0,0.0);
#    glPointSize(5);
    glVertex2f(-3.0,9.0);
    glVertex2f(2.0,6.0);
    glVertex2f(3.0,8.0);
    glVertex2f(-2.5,6.5);
    glEnd();
    #画闭折线
    glBegin(GL_LINE_LOOP);
    glColor3f(0.0,1.0,1.0);
    glVertex2f(7.0,7.0);
    glVertex2f(8.0,8.0);
    glVertex2f(9.0,6.5);
    glVertex2f(10.3,7.5);
    glVertex2f(11.5,6.0);
    glVertex2f(7.5,6.0);
    glEnd();
    #画填充多边形
    glBegin(GL_POLYGON);
    glColor3f(0.5,0.3,0.7);
    glVertex2f(-7.0,2.0);
    glVertex2f(-8.0,3.0);
    glVertex2f(-10.3,0.5);
    glVertex2f(-7.5,-2.0);
    glVertex2f(-6.0,-1.0);
    glEnd();
    #画四边形
    glBegin(GL_QUADS);
    glColor3f(0.7,0.5,0.2);
    glVertex2f(0.0,2.0);
    glVertex2f(-1.0,3.0);
    glVertex2f(-3.3,0.5);
    glVertex2f(-0.5,-1.0);
    glColor3f(0.5,0.7,0.2);
    glVertex2f(3.0,2.0);
    glVertex2f(2.0,3.0);
    glVertex2f(0.0,0.5);
    glVertex2f(2.5,-1.0);
    glEnd();
    #画连接四边形
```

```
    glBegin(GL_QUAD_STRIP);
    glVertex2f(6.0,-2.0);
    glVertex2f(5.5,1.0);
    glVertex2f(8.0,-1.0);
    glColor3f(0.8,0.0,0.0);
    glVertex2f(9.0,2.0);
    glVertex2f(11.0,-2.0);
    glColor3f(0.0,0.0,0.8);
    glVertex2f(11.0,2.0);
    glVertex2f(13.0,-1.0);
    glColor3f(0.0,0.8,0.0);
    glVertex2f(14.0,1.0);
    glEnd();
    #画三角形
    glBegin(GL_TRIANGLES);
    glColor3f(0.2,0.5,0.7);
    glVertex2f(-10.0,-5.0);
    glVertex2f(-12.3,-7.5);
    glVertex2f(-8.5,-6.0);
    glColor3f(0.2,0.7,0.5);
    glVertex2f(-8.0,-7.0);
    glVertex2f(-7.0,-4.5);
    glVertex2f(-5.5,-9.0);
    glEnd();
    #画连续三角形
    glBegin(GL_TRIANGLE_STRIP);
    glVertex2f(-1.0,-8.0);
    glVertex2f(-2.5,-5.0);
    glColor3f(0.8,0.8,0.0);
    glVertex2f(1.0,-7.0);
    glColor3f(0.0,0.8,0.8);
    glVertex2f(2.0,-4.0);
    glColor3f(0.8,0.0,0.8);
    glVertex2f(4.0,-6.0);
    glEnd();
    #画扇形三角形
    glBegin(GL_TRIANGLE_FAN);
    glVertex2f(8.0,-6.0);
    glVertex2f(10.0,-3.0);
    glColor3f(0.8,0.2,0.5);
    glVertex2f(12.5,-4.5);
    glColor3f(0.2,0.5,0.8);
    glVertex2f(13.0,-7.5);
    glColor3f(0.8,0.5,0.2);
    glVertex2f(10.5,-9.0);
    glEnd();
glutInit();
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutInitWindowSize(500,500);
glutInitWindowPosition(0,0);

glutCreateWindow("Geometric Primitive Types");

init();
glutDisplayFunc(RenderScene);

#glutReshapeFunc(RenderScene);
#glutDisplayFunc(chaingeSize);

glutMainLoop();
```

4.2. 题目二

```
#coding:utf8
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
from math import *
```

```
def init():
    glClearColor(1.0,1.0,1.0,1.0);
    gluOrtho2D(0,500,0,500)
#    glClear(GL_COLOR_BUFFER_BIT);

def MidPointLine(x0,y0,x1,y1):
    glClear(GL_COLOR_BUFFER_BIT);
    dx=x1-x0;
    dy=y1-y0;
    d=dx-2*dy;#初始化判别式
    incrE=-2*dy;#取像素E判别式增量,取下
    incrNE=2*(dx-dy);#取上

    glLineWidth(2.0);
    glBegin(GL_LINES);
    glColor3f(1.0,0.0,0.0);
    glVertex2f(x0,y0);
    x=x0;y=y0;

    while x<x1:
        if d>0:
            d=d=incrE;
        else:
            d=d+incrNE;
            y=y+1;

        x=x+1;
        print x,y
        glVertex2f(x*1.0,y*1.0);
    glEnd();
    glFlush();


def drawline():
    MidPointLine(20,20,500,300);

glutInit();
glutInitDisplayMode(GLUT_RGB|GLUT_SINGLE);
glutInitWindowSize(500,500);
glutCreateWindow("MidPointLine");
init();
glutDisplayFunc(drawline);
glutMainLoop();
```