# 《计算机图形学》系统设计11月进展报告

**张帅 151220162**

目录

[《计算机图形学》系统设计11月进展报告 1](#_Toc499838030)

[概要 2](#_Toc499838031)

[1 算法原理介绍 2](#_Toc499838032)

[1.1 中点椭圆生成算法 2](#_Toc499838033)

[1.2 多边形扫描填充算法 2](#_Toc499838034)

[2 系统框架设计 3](#_Toc499838035)

[2.1 实验环境 3](#_Toc499838036)

[2.2 系统架构 3](#_Toc499838037)

[3 系统功能介绍及使用说明 4](#_Toc499838038)

[3.1 直线绘制+编辑 5](#_Toc499838039)

[3.2 圆绘制&填充+编辑 5](#_Toc499838040)

[3.3 椭圆绘制+编辑 6](#_Toc499838041)

[3.4 多边形绘制&填充+编辑 7](#_Toc499838042)

## 概要

本图形绘制系统采用面向对象设计，以Qt和OpenGL作为基础，目前已实现绘制、填充及编辑功能，可以绘制的图形包括直线、圆、椭圆、多边形，可以填充的图形包括圆和多边形。使用Qt提供界面交互，使用OpenGL提供绘制，界面简洁美观，操作方便。

本月新增功能：

1. 椭圆绘制、圆和多边形的填充，其中椭圆绘制使用了中点椭圆生成算法，多边形填充使用了多边形扫描填充算法，圆的填充算法为自写；

2. 将原先工程使用Qt重新实现，添加了更加用户友好的交互界面；

3. 添加了图形编辑功能。

## 1 算法原理介绍

### 1.1 中点椭圆生成算法

将椭圆按照象限分为4个部分，先绘制出第一象限的部分，然后根据对称性绘制出其余3个象限的部分。

第一象限的部分分为两个区域：区域1：切线斜率小于1；区域2：切线斜率大于1.

区域1：

决策参数：.

初始取值：.

更新公式：.

区域2：

决策参数：

初始取值：. (其中()为区域1最后一个点

更新公式：.

### 1.2 多边形扫描填充算法

先后使用了有序边表与活化边表，得到各扫描行的填充范围，从而进行填充。

边表使用了如下数据结构：

|  |
| --- |
| struct Edge  **{**  Edge**()** **{}**  Edge**(**double xi**,** double dx**,** int ymax**)**  **{**  **this->**xi **=** xi**;**  **this->**dx **=** dx**;**  **this->**ymax **=** ymax**;**  **}**  double xi**; //本扫描行该边的起始点横坐标**  double dx**; //该边y每增加1，x的增加量，即斜率的倒数**  int ymax**; //该边最高点的y值**  **};** |

a. 生成有序边表：扫描多边形中所有边，跳过所有点y值相同的水平边，将所有边最下方的点加入到有序边表中，填入对应的xi，根据该边起始点计算出dx。若该点在上下侧，则ymax为该边的最高点y值；若该点在左右两侧，则ymax为该边最高点y值-1（为了防止重复绘制顶点带来的内外侧判断错误）。

b. 根据有序边表生成活化边表：扫描当前扫描线及以下的所有有序边表，将ymax大于等于当前y值的Edge项加入到本扫描线对应的活化边表中，加入时xi需要根据当前y值和有序边表中的xi，dx计算，其余项保持不变。然后根据xi对当前扫描线的Edge项从小到大排序。

c. 根据活化边表生成各扫描行填充范围：遍历当前扫描行所有Edge项，两两扫描，从索引为偶数的项开始，填充到下一个项的xi为止。

## 2 系统框架设计

### 2.1 实验环境

操作系统：Windows 10

编程语言：C++11

开发环境：Qt Creator，Qt 5.4

### 2.2 系统架构

用Qt向用户提供交互，用OpenGL提供底层绘制，考虑到Qt与OpenGL的兼容性，使用了继承自QWidget的子类QGLWidget实现Qt环境下的OpenGL绘制。

QMainWindow负责与用户交互，并以标签页的形式集成了QGLWidget；

QGLWidget负责绘制本标签页的内容，并集成了各类FigureControl，用于将本标签页的交互信号送给不同的图形控制类；

FigureControl负责该类型图形的交互，有LineControl、CircleControl、EllipseControl、PolygonControl等，用于分别提供不同图形的不同交互方式；

Figure负责保存该图形的数据信息，包括顶点、圆心、半径以及轮廓点、填充点等，集成了Point类，图形绘制通过调用Point的draw函数完成；

Point负责实现绘制单个普通点或标记点，并向上提供其他常用功能函数支持。

系统核心架构如下图：

QMainWindow

QGLWidget

FigureControl

Figure

Point

界面交互：管理多个QGLWidget的标签页，接收键盘鼠标信息

标签页绘制：管理FigureControl的子类，实现本标签页的绘制

单图形管理：管理该图形的Figure类，向上提供该种图形的绘制接口

图形数据：管理本图形数据，提供图形绘制、修改、获取数据的接口

点信息：作为Figure的基础，提供绘制、比较、获取信息等接口

相比于上个月，图形类架构略有改进。改进后的继承关系如下：

Figure

SimpleFigure

Polygon

Line

Circle

Ellipse

Area

其中，Figure类为一个抽象类，是所有图形的基类；Area类为所有可填充图形的基类，提供图形填充的接口；SimpleFigure为直接由点组成的简单几何图形的类；Polygon类内聚集了Line，故不是直接由点组成的图形。

## 3 系统功能介绍及使用说明

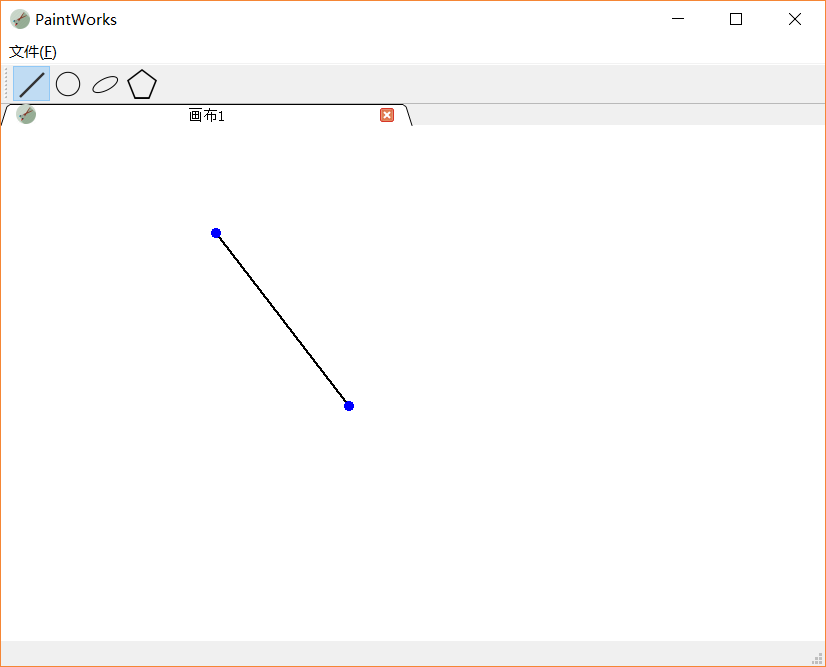
本系统可以绘制4种图形：直线、圆、椭圆、多边形，可以实现对圆和多边形的填充。图形相互之间如有覆盖，则按照绘制顺序进行覆盖。绘制过程中会以标记的形式显示当前正在绘制的图形，绘制完成后可以拖动蓝色的标记点对刚完成绘制的图形进行修改。

进入系统之后，“文件”中点选“新建画布”或按快捷键Ctrl+N即可开始绘制，点击上方按钮（或按快捷键Alt+L/C/E/P）可以选择将要绘制的图形。

本报告中所述功能**均已实现**。

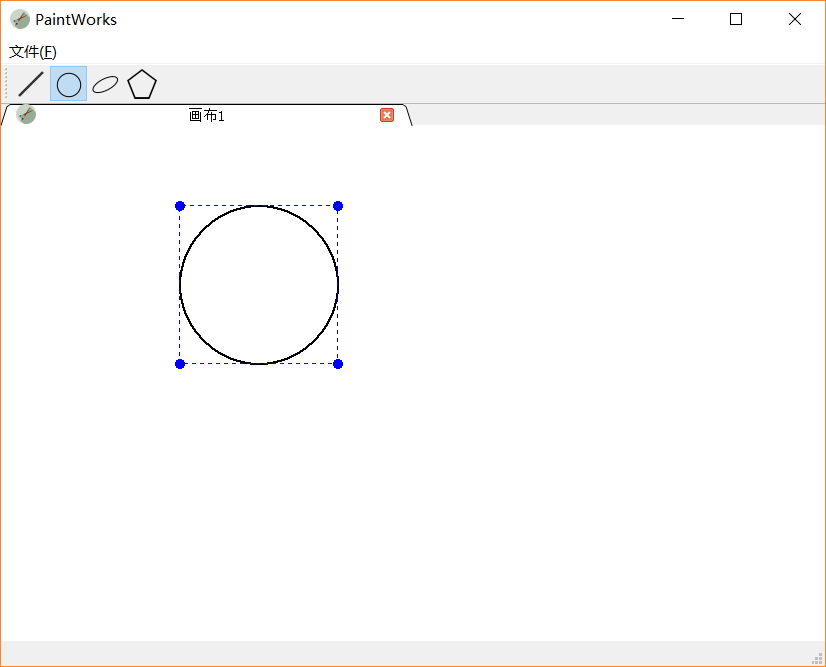
### 3.1 直线绘制+编辑

直线绘制只需用鼠标点选即可。鼠标左键在直线起始点按下，按住不放，直到移动到直线终点后放开鼠标左键即可完成绘制。在拖动过程中可以看到直线的实时显示。如图：

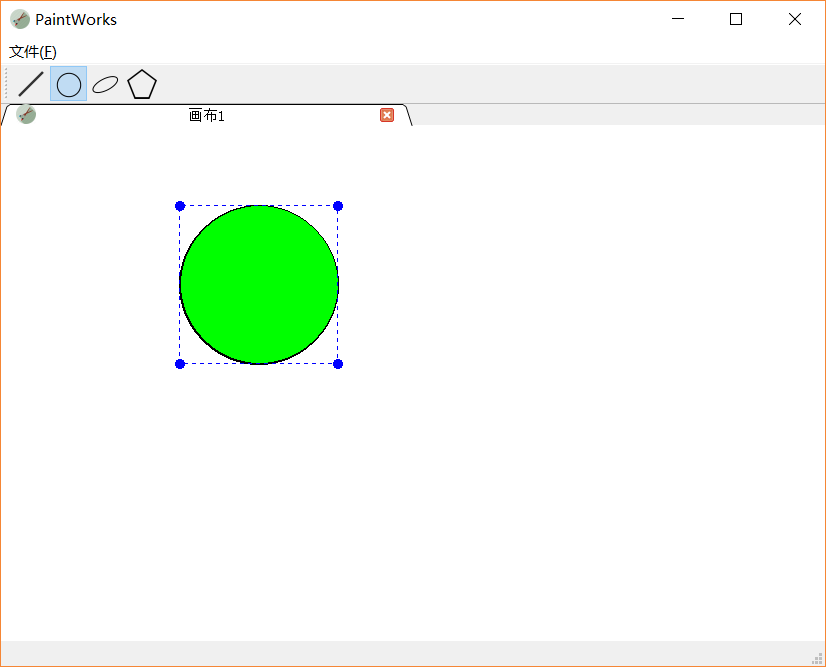


### 3.2 圆绘制&填充+编辑

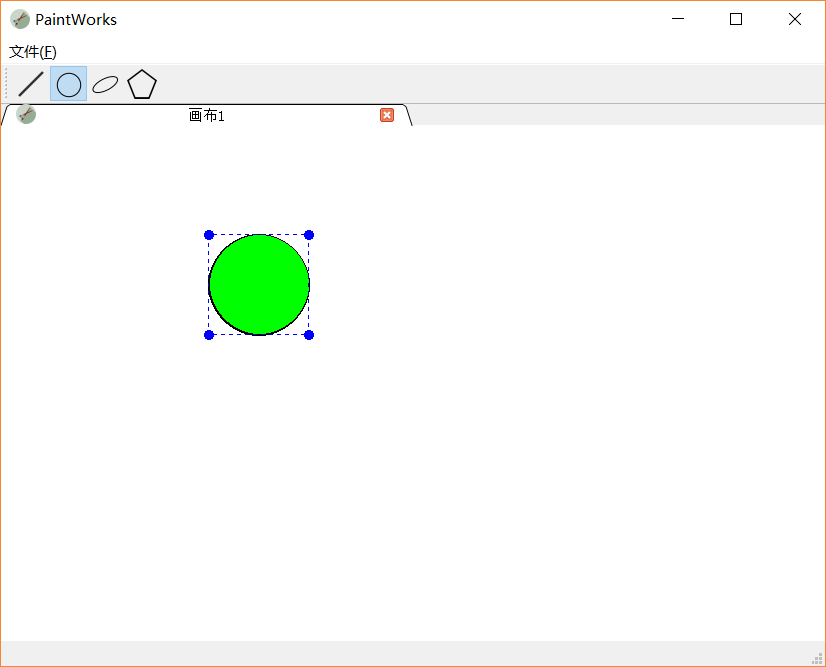
圆的绘制通过鼠标点选圆心+拖动半径完成。鼠标左键点下后拖动即可完成圆的绘制。如图：



按快捷键F即可实现填充：

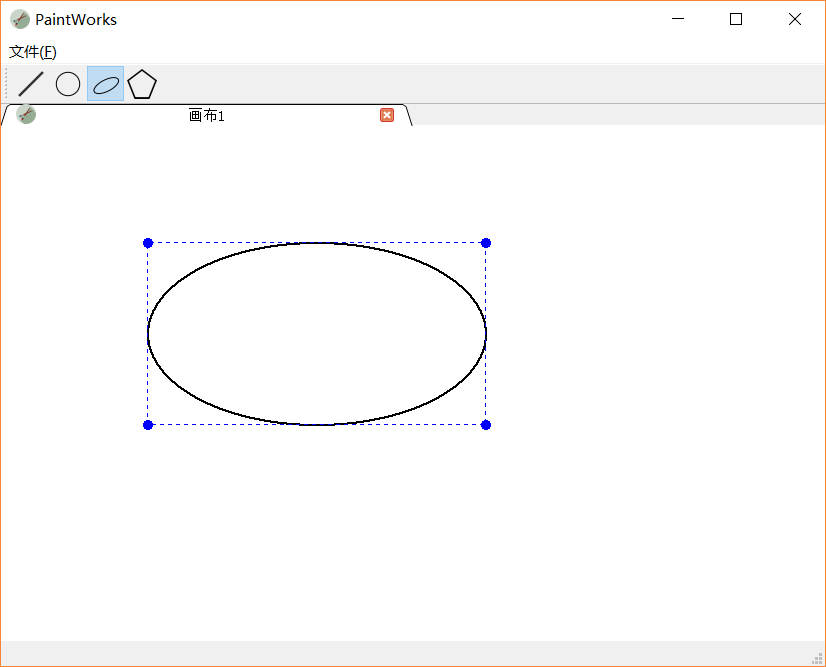


拖动蓝色标记点可以对圆进行编辑，实时更新填充点：



### 3.3 椭圆绘制+编辑

椭圆绘制也是通过点击+拖动完成，同样完成之后可以拖动蓝色点进行更改，如图：



### 3.4 多边形绘制&填充+编辑

多边形绘制只需用鼠标点选多个点即可。在起始点位置用鼠标左键点击一下，绘制第一条边，再次点击之后自动开始绘制第二条边，通过同样的方式添加多条边，最终在起始点位置点一下将折线闭合即可完成一个多边形的绘制。效果如下图：

这里考虑到准确点到起始点可能较为困难，做了一些细节调整，点在起始点附近便会自动将终点连到起始点位置。

