CG_1_Minidraw 实验报告

PB21010479　王曹励文

2024 年 3 月 1 日

1　问题叙述

写一个交互式画图小程序 MiniDraw，要求画直线 (Line)，椭圆 (Ellipse)，矩形 (Rectangle)，多边形 (Polygon) 等图形元素（图元），并尽可能满足以下的要求：

1. 每种图元需用一个类（对象）来封装，如 Line，Ellipse，Rect，Polygon，Freehand（自由绘制模式，optional），
2. 各种图元从一个父类来继承，如 Shape，
3. 使用类的多态调用绘图函数，
4. 画图工具的拓展功能（线条粗细颜色、形状填充、对象选取、对象变换、顶点编辑等，optional）.

2　操作方法与实验结果

2.1　初始界面与图片插入

如图 1 所示为初始界面.

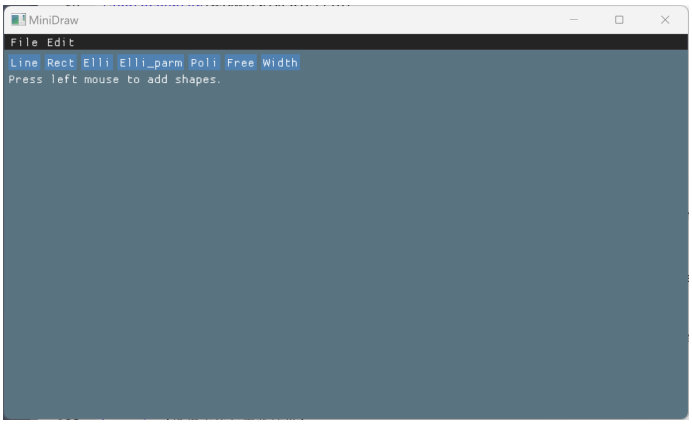
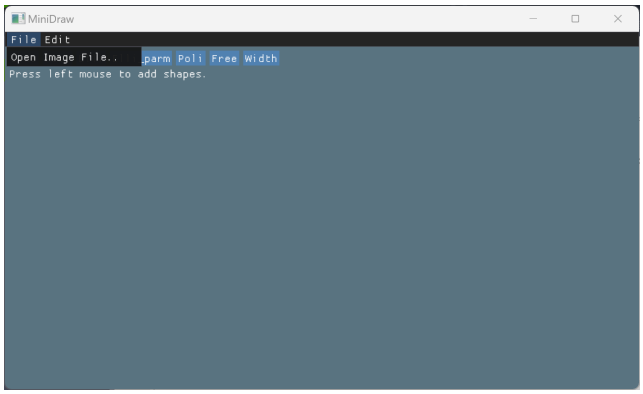
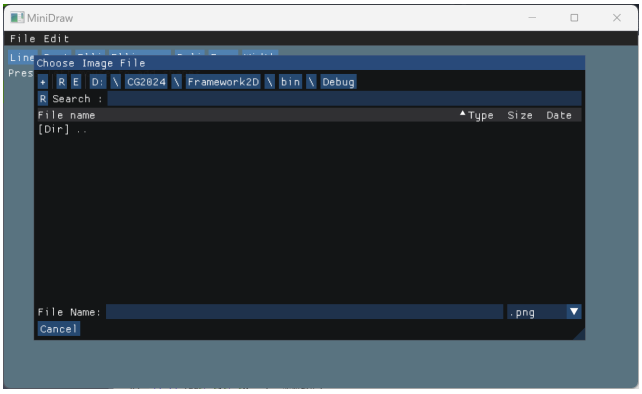


图 1: 初始界面

点击”file” 按钮，表示向其中插入一张图片，会出现 “Open Image File” 按钮，如图 2(a) 所示，点击后出现”Choose Image File” 对话框，如图 2(b) 所示，选择文件后点击 ok，即可将图片插入界面，插入后的结果如图 3(b) 所示，插入时可以选择文件的类型，如图 3(a) 所示.

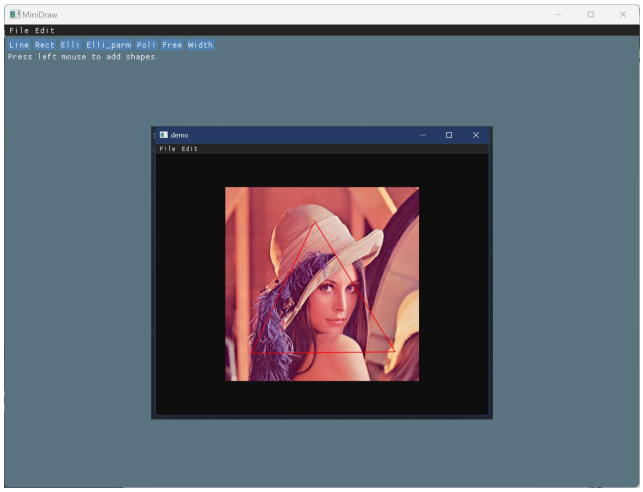


(a) “Open Image File”

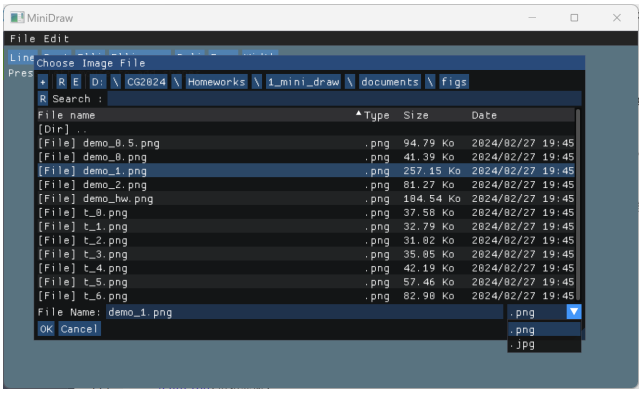


(b) ”Choose Image File”

图 2: 插入图片



(a) 插入图片示例

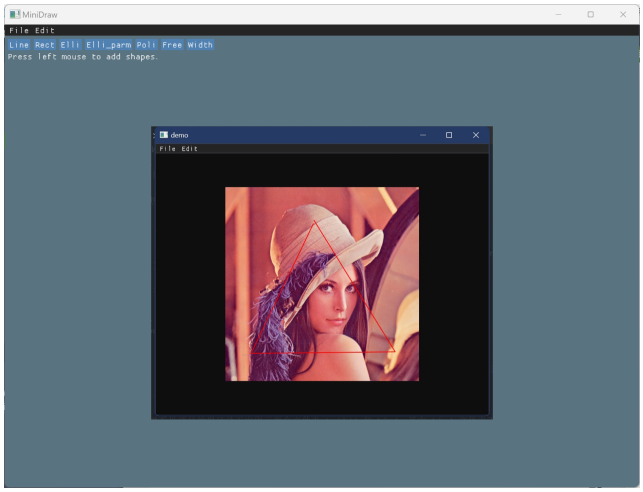


(b) 切换图片格式

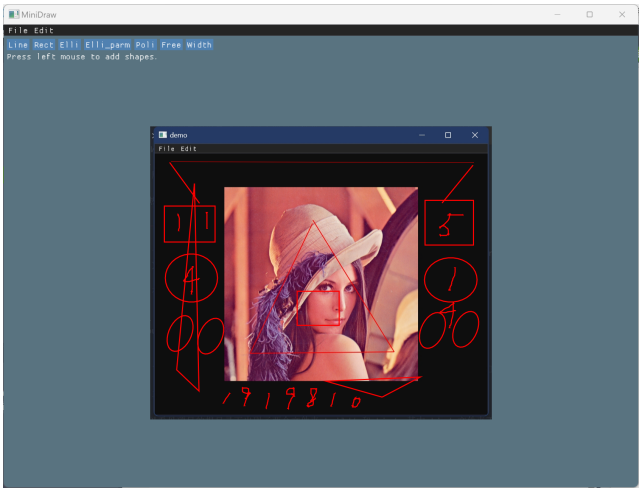
图 3: 插入图片结果

点击”Edit” 按钮，弹出”Enable Canvas” 窗口，默认该选项不打开，如图 4(a) 所示. 打开后，可以实现在该图形上进行图元的绘制，如图 4(b) 所示，具体的图元在 2.2 中介绍.

事实上，已实现的功能有，插入多张图片，每一次插入图片后，前一次绘制的图元不再展示.



(a) "Enable Canvas"



(b) 图元的绘制示例

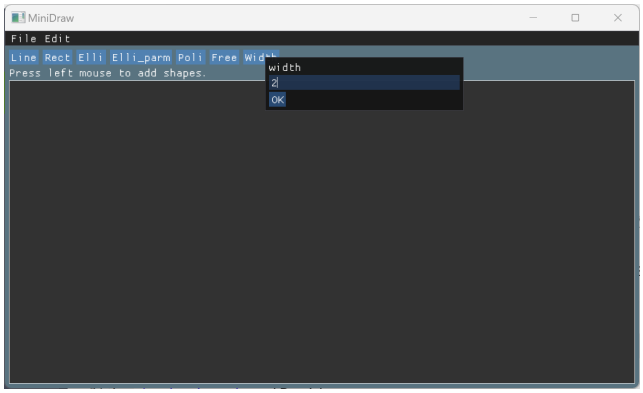
图 4: 图元绘制结果

2.2 图元的绘制

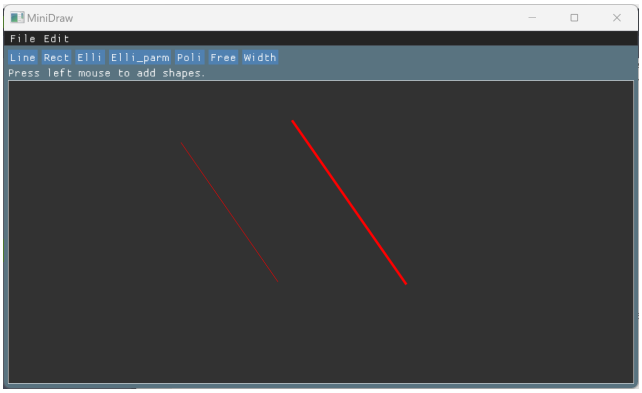
如图 1 所示，Line 元素代表直线的绘制，Rect 元素代表矩形的绘制，Elli 元素代表椭圆的绘制，Elli_parm 元素代表带有参数的椭圆的绘制，Poli 代表多边形的绘制，Free 代表自由曲线状态，Width 代表宽度. 具体每一部分的使用请参考如下说明，以下的说明均默认"Enable Canvas" 开启.

2.2.1 Width

点击 Width，弹出的界面 width 中可以设置线条的宽度，写入 float 类型值，如图 5(a) 所示. 图 5(b) 展示了 width 设置为 1 和 5 的两种线条.



(a) width 界面



(b) width 设置为 1 和 5 的两种线条

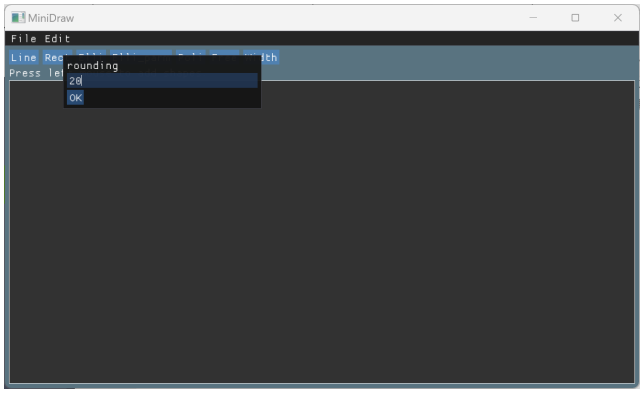
图 5: Width

2.2.2 Line

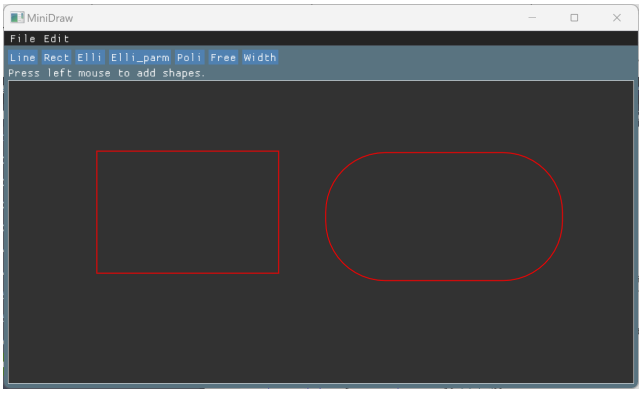
点击 Line，点击一次设立起始点，拖动鼠标实时更新绘制的线段，再次点击确定线段的终点. 绘图示例如图 5(b) 所示.

2.2.3 Rect

点击 Rect，弹出 rounding 窗口，如图 6(a) 所示. 输入的值为正方形的角度变化值，值为 0 时代表矩形，值为 0 和 120 的示例如图 6(b) 所示. 绘制时，点击一次设立左上起始点，拖动鼠标实时更新绘制的图元形状，再次点击确定右下的终点.



(a) rounding 窗口



(b) rounding 设置为值为 0 和 120 的示例

图 6: Rect

2.2.4 Elli

点击 Elli，弹出 angle 窗口，如图 7 所示. 输入的值为正方形的角度变化值，值为 0 时代表无角度，值为 0 和 5 的示例如图 7 所示. 绘制时，点击一次设立左上起始点，拖动鼠标实时更新绘制的图元形状，再次点击确定右下的终点.

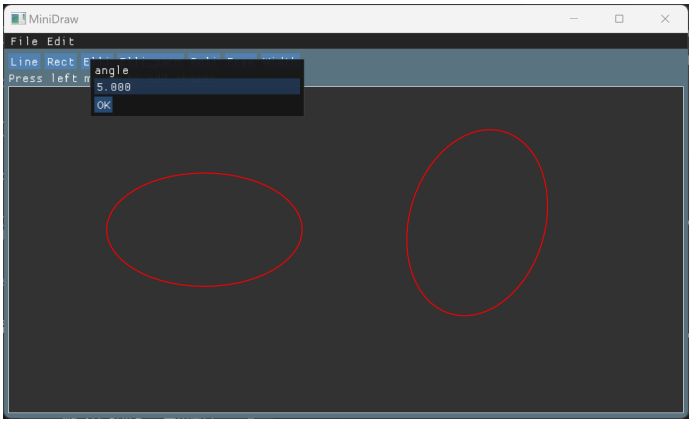


图 7: Elli

2.2.5 Elli_parm

与上面 Elli 的实现不同的是，这里描述了给出长短半轴和旋转角度的方法. 点击 Elli_parm，弹出参数窗口，radius_x, radius_y, angle 分别表示长短半轴和旋转角度. 之后单点即可出现对应参数的椭圆，结果如图 8 所示.

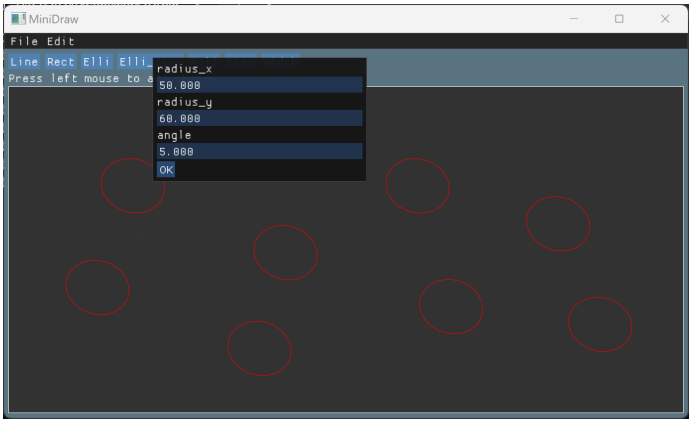


图 8: Elli_parm

2.2.6 Poli

左键点击 Poly，即可进入多边形模式，左键创建新的点，并且可以按顺序连接已经有的点，右键则连接起点与终点构成闭合，结果如图 9 所示.

图 9: Poly

2.2.7 Free

点击 Free，即可实现自由画图模式，如图 10 所示为不同宽度的实现效果.

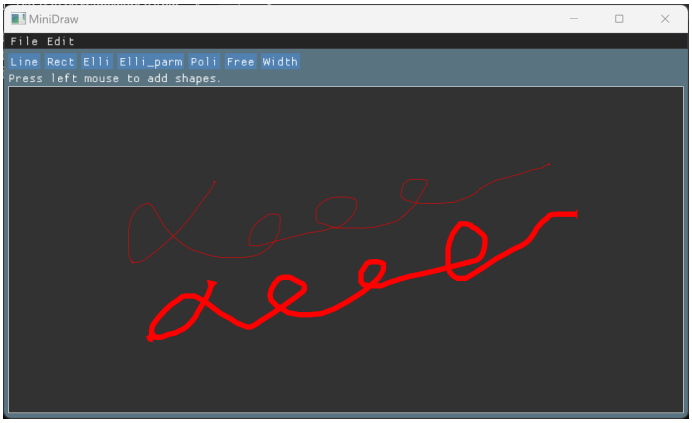


图 10: Free

2.2.8 总结

基于以上实现的功能，我们可以绘制部分图像，示例如图 11 所示. 笔者发现当宽度较长时，椭圆，矩形等可以实现类似的填充效果.

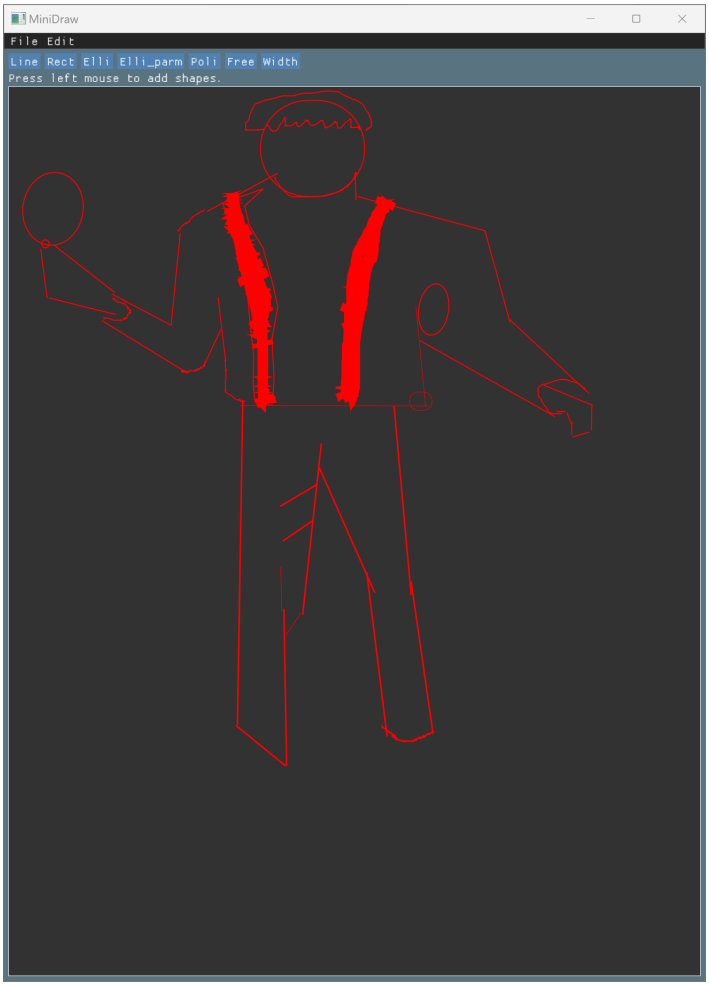


图 11: 综合图形示例

3 实验原理

本实验的核心目的之一是理解类的封装，继承与多态. 多种不同类型图元作为 Shape 类的子类，其关系如图 12 所示.

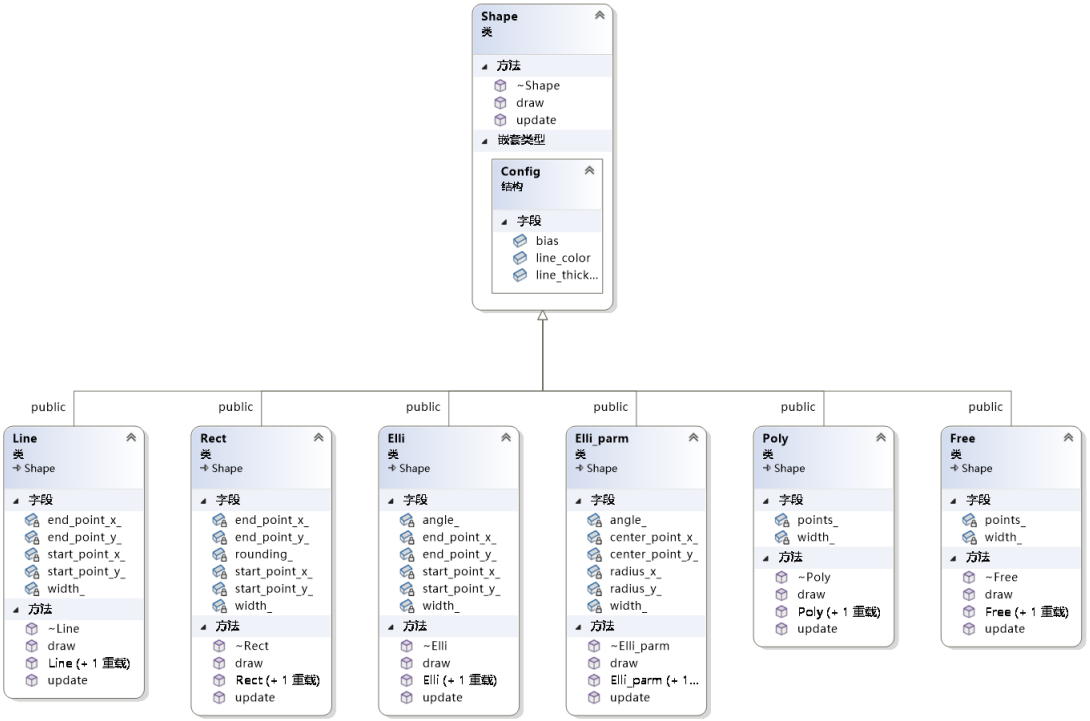


图 12: Shape 类与子类关系

最初实现的 Image 插入图像类与 Canvas 画布类继承于 Component 类，其关系如图 13 所示.

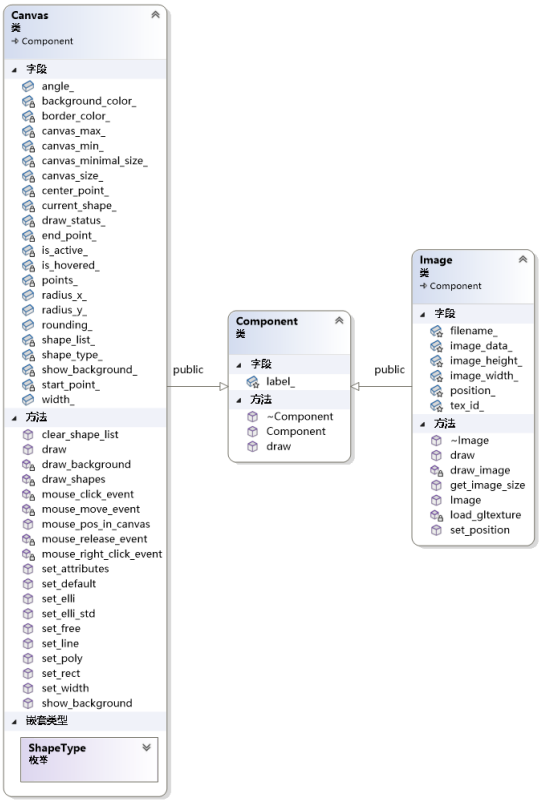


图 13: Image 类与 Component 类

4 反思与总结

本次实验的核心目的是学习类的封装、继承与多态，这些是 C++ 语言特有的. 由于个人精力和时间与能力受限，虽然已经实现了调整图元绘制粗细的功能，并通过其近乎实现了填充效果；实现了 Freehand 模式，在 Rect 和 Elli 模式都给予了适当的参数，实现了其旋转的功能；实现了添加图片并且在图片上进行绘制的功能（我认为这个功能是重要的，可以导入图片就进行描边，以确定某些图的边界形状）.

我认为可以进行如下方向的改进：

1. 实现图元的选择与删除，这是由于很难一步绘制到位. 选择我的想法是寻找库的内置函数，或者判断到图元的距离是否小于某一个小量.

2. 实现颜色界面的调整. 目前只能使用原先给出的默认颜色，需要添加颜色选择窗口并且将参数直接传给图元类，再调用已经存在的函数.

3. 实现更强的交互性. 如选择点之后对已有图元进行操作，改变大小等.