1. 光栅化

欢迎大家来到《实时三维图形基础》课程的第一次作业！

作业说明

本作业旨在让大家亲手体会图形学中光栅化的过程。光栅化（Rasterization）就是把顶点数据转换为片元的过程，片元中的每一个元素对应于帧缓冲区中的一个像素，是一种将几何图元变为二维图像的过程。

本作业基于 OpenGL 开发，OpenGL（Open Graphics Library）是一个跨编程语言、跨平台的编程图形程序接口，它将计算机的资源抽象称为 OpenGL 的对象，对这些资源的操作抽象为 OpenGL 指令。

作业提交

- https://bhpan.buaa.edu.cn/link/AA43F0DCA76D9740269BE92EC8EF130D0A，提交至“第一次作业”子目录下。
- 文件名：学号-姓名-第一次作业.zip
- 提交内容：
    - 实验报告（实现思路、效果展示）
    - rasterization.cpp（如果有其他修改一并提交）

作业要求

- 补全画图小程序，学习光栅化相关知识：
    - 实现 Bresenham 直线算法
    - 实现椭圆形的光栅化
- 学习着色器、渲染管线等 OpenGL 基础知识。

`src/rasterization.cpp` 是这次作业中需要修改的文件，本次作业要求完成 `draw_line_bresenham` 和 `draw_ellipse` 两个函数（可以参考 `draw_line_dda` 函数的实现方式）。

项目结构

本项目目录包含以下若干子目录：

- `src/` - 本次作业项目的源代码：
    - 本项目从 `main.cpp` 开始运行；
    - shader 目录下是顶点着色器和片段着色器的文本文件，`shader.h` 和 `shader.cpp` 是自定义的着色器类，方便进行着色器的调用；
    - `utils.h` 和 `utils.cpp` 定义了两个工具类，`Vector2f` 类封装了长度为2的浮点向量，Pixel 类代表屏幕显示的像素点；
    - `rasterization.h` 和 `rasterization.cpp` 包括3个用于实现光栅化的函数。
- `deps/` - 本次作业的第三方依赖文件，包含 `glfw`，`glad` 和 `imgui` 三部分。不要随意修改和移动这部分的文件。

补充阅读

1. Bresenham 算法 The Bresenham Line-Drawing Algorithm (helsinki.fi)。
2. OpenGL 3.3 (Core Profile) - March 11, 2010 (khronos.org) 3.5.1节光栅化标准。