计算机学院实验报告

|  |  |  |
| --- | --- | --- |
| 实验题目： | | 学号：202300130183 |
| 日期：2025/ | 班级：23级智能班 | 姓名：宋浩宇 |
| **Email：[2367651943@qq.com](mailto:2367651943@qq.com)**  **202300130183@mail.sdu.edu.cn** | | |
| 实验目的：  把实验要求复制上来 | | |
| 实验环境介绍：  软件环境：  主系统：Windows 11 家庭中文版23H2 22631.4317  虚拟机软件：Oracle Virtual Box 7.1.6  虚拟机系统：Ubuntu 18.04.2 LTS  编辑器：Visual Studio Code  编译器：gcc 7.3.0  计算框架：Eigen 3.3.7  硬件环境：  CPU：13th Gen Intel(R) Core(TM) i9-13980HX 2.20 GHz  内存：32.0 GB (31.6 GB 可用)  磁盘驱动器：NVMe WD\_BLACKSN850X2000GB  显示适配器：NVIDIA GeForce RTX 4080 Laptop GPU | | |
| 解决问题的主要思路：  简单介绍本实验的所需知识点（如需要学习的API、算法等）、算法思路、额外讨论的要点/亮点**（助教会仔细看此块内容）**  例如：  本实验的主要思路如下：   1. 学习基本的OpenGL/GLUT接口，熟悉reshape，display，keybord，mouse，motion五种基本的GLUT函数配置。 2. 学习光栅图形学的直线段的扫描转换算法：数值微分法（DDA）和Bresenham算法，掌握其算法的多种不同实现方法，并讨论**斜率的不同情况**对算法的影响。 3. 学习光栅图形学的圆弧的扫描转换算法：中点画圆法，利用圆的**八分对称性**实现高效算法，并讨论**中点画圆法的分辨率**（像素点的个数）。 | | |
| 实验步骤与实验结果：  分一、二、三……列举实验步骤，每一步一般先介绍算法内容，再介绍代码实现（代码块可以从VSCode里直接复制，就有代码渲染了），最后贴图展示实验结果。内容**不能太少**，但**也不用太多**（**实验不用卷**）。  例如：   1. 直线段的扫描转换算法：DDA和Bresenham算法 2. **DDA算法**   DDA算法是基于微分思想的一种朴素算法。对于起点 、终点 的直线，其斜率 。我们对每个 找到一个合适的 ，使得尽可能接近这条直线。事实上，从 开始，每次 累加1（或-1，取决于方向）， 累加 （或 ，取决于方向）。为了寻找到和 最接近的整数点，我们取 作为像素点。  为什么我们要使用累加？因为在循环过程中，如果不适用累加的形式，就需要进行相同次数的乘法，这是得不偿失的。  上述算法对**斜率**有要求。如果斜率 ，那么此时图像会变成什么样？容易想象，此时会有多个纵坐标不会被画到。对于横坐标移动距离为 、纵坐标移动距离为 的直线，我们当然希望图像的分辨率更高，即像素点的个数为 。因此，当斜率 时，应当将 在运算中的地位互换。  那么，有没有一种简单的是实现方法，以避免这种互换呢？显然，如果斜率 ，那么 的累加步长就不是1了，而是 。因此，我们只需要根据斜率合理设置步长，然后从起点到终点依次画点即可。  下面是代码实现：   |  | | --- | | // DDA画线算法  void DDALine()  {  // 绘制每一条线  for (int i = 0; i < numL; i++)  {  glBegin(GL\_POINTS); // 定义一个原始顶点  glColor3f(0.0, 1.0, 0.0); // 设置颜色模式  // 输入起点终点  int x1 = line[i][0]; int y1 = line[i][1];  int x2 = line[i][2]; int y2 = line[i][3];  float x = x1, y = y1;  // 计算步数  int s = (abs(x2 - x1) >= abs(y2 - y1)) ? abs(x2 - x1) : abs(y2 - y1);  // 计算步长  double dx = float(1.0 \* (x2 - x1) / s);  double dy = float(1.0 \* (y2 - y1) / s);  // 画点  for (int i = 0; i < s; i++)  {  glVertex2d(x, int(y + 0.5));  x += dx; y += dy;  }  glEnd();  }  } |   实验效果： | | |
| 实验中存在的问题及解决：  列举1~3个实验中的关键问题和解决方法**（助教会仔细看此块内容）**  例如：  问题1：直线的绘制过程中，如果不讨论斜率的大小，会发生什么情况？  答1：如果斜率的绝对值大于1，而仍然按照 增量1的形式画点，将会导致点的数量十分稀疏。  问题2：中点画圆法为什么使用八分对称性？  答2：将斜率的情况归结为一种，避免讨论。  问题3：DDA/Bresenham直线扫描算法和中点画圆法的分辨率（即像素点的个数）各为多少？  答3：直线扫描算法的分辨率为 。中点画圆法的每个圆弧的像素点均为 ，即分辨率为 。 | | |