《计算机图形学》系统技术报告（原理以及性能测试）

作者姓名 李东昊 141220051

联系方式: 961171432@qq.com 15905192178

(南京大学 计算机科学与技术系, 南京 210093)

摘 要: 在 OpenGL 开发环境下，实现一个较为简易的图形处理系统，完成简单的图形绘制以及一些处理功能，比如图形裁剪、平移、旋转、缩放、存储等功能。

关键词: OpenGL；计算机图形学；图形处理

1 原理

1.1 图形绘制

1.1.1 直线

直线绘制实现了 DDA 算法以及 Bresenham 算法

因为系统实现过程主要使用 DDA 算法，所以这里只进行 DDA 算法的介绍

1. 输入直线的起点、终点；

2. 计算 x 方向的间距：△X 和 y 方向的间距：△Y。

3. 确定单位步进，取 MaxSteps = max（△X，△Y）；若△X>=△Y，则 X 方向的步进为单位步进，X 方向步进一个单位，Y 方向步进△Y/MaxSteps;否则相反。

4. 设置第一个点的像素值

5. 令循环初始值为 1，循环次数为 MaxSteps，定义变量 x,y，执行以下计算：

a. x 增加一个单位步进，y 增加一个单位步进

b. 设置位置为（x,y）的像素值

1.1.2 曲线

采用 B 样条（均匀）曲线绘制

B样条曲线的定义

B样条曲线分为均匀B样条曲线和非均匀B样条曲线，这里只讨论均匀B样条曲线。给定n+1个控制点Pi(i=0,1,2,...,n)的坐标Pi，n次B样条曲线段的参数表达式为

$$P(t)=\sum_{t=0}^{n}P_iF_{i,n}(t),t\in[0,1]$$

式中为n次B样条基函数，其形式为

$$F_{i,n}(t)=\frac{1}{n!}\sum_{j=0}^{n-i}(-1)^jC_{n+1}^{j}(t+n-i-j)^n$$

其中

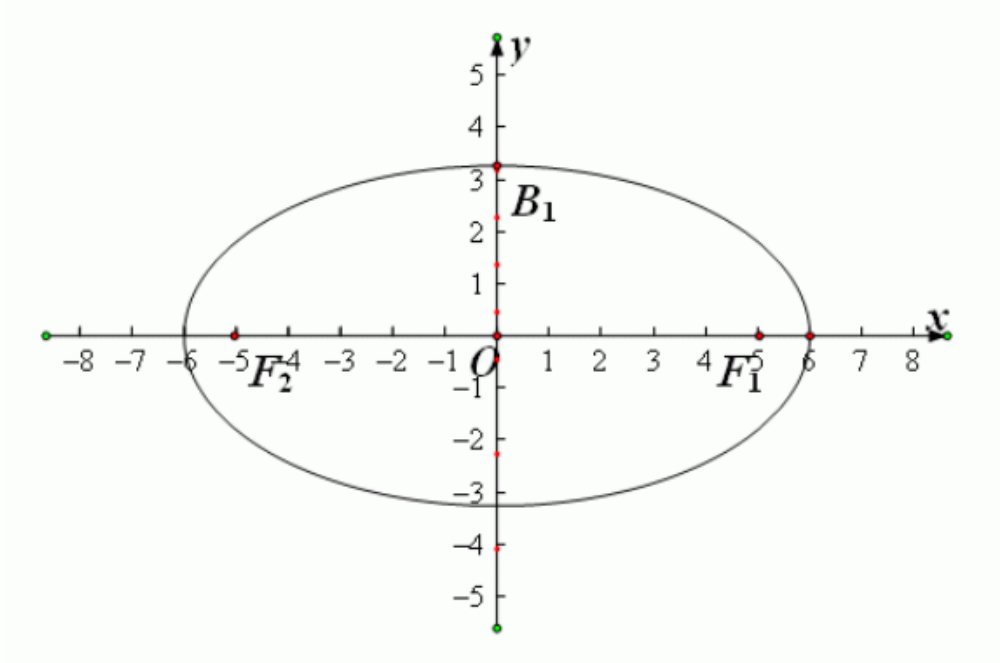
$$C_{n+1}^{j}=\frac{(n+1)!}{j!(n+1-j)!}$$

1.1.3 多边形

多边形使用之前实现的 DDA 算法进行取点连接

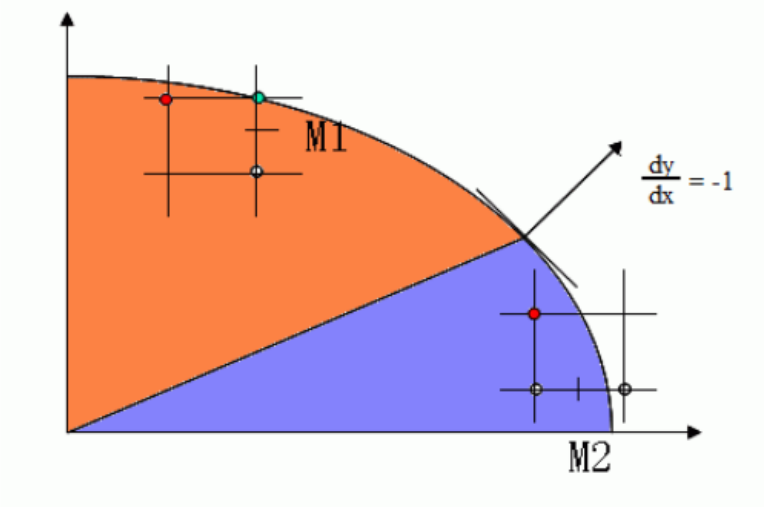
1.1.4 椭圆/圆形

中点生成算法绘制



图（1）椭圆的对称性

对于中点画圆弧算法，要区分出椭圆弧上哪段 Δx 增量变化显著，哪段 Δy 增量变化显著，然后区别对待。由于椭圆的对称性，我们只考虑第一象限的椭圆圆弧，如图（2）所示：



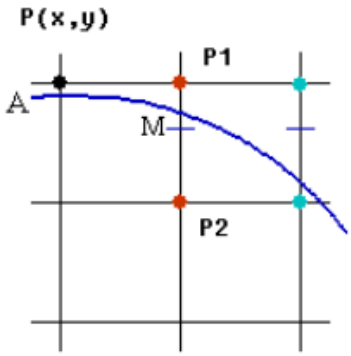
图（2）第一象限椭圆弧示意图

定义椭圆弧上某点的切线法向量 N 如下：

$$N(x,y)=\frac{\partial F(x,y)}{\partial x}i+\frac{\partial F(x,y)}{\partial y}j$$
$$=2b^2xi+2a^2yj$$

对方程 1 分别求 x 偏导和 y 偏导，最后得到椭圆弧上(x,y)点处的法向量是（2b2x，2a2y）。dy/dx = -1 的点是椭圆弧上的分界点。此点之上的部分（橙褐色部分）椭圆弧法向量的 y 分量比较大，即：2b2(x + 1) < 2a2(y - 0.5)；此点之下的部分（蓝紫色部分）椭圆弧法向量的 x 分量比较大，即：2b2(x + 1) > 2a2(y - 0.5)。

对于图（2）中橙褐色标识的上部区域，y 方向每变化 1 个单位，x 方向变化大于一个单位，因此中点算法需要沿着 x 方向步进画点，x 每次增量加 1，求 y 的值。同理，对于图（2）中蓝紫色标识的下部区域，中点算法沿着 y 方向反向步进，y 每次减 1，求 x 的值。先来讨论上部区域椭圆弧的生成，如图（3）所示：



图（3）中点画椭圆算法对上部区域处理示意图

假设当前位置是 P(xi, yi)，则下一个可能的点就是 P 点右边的 P1(xi+1, yi)点或右下方的 P2(xi+1, yi-1)点，取舍的方法取决于判别式 di，di 的定义如下：

di = F(xi+1, yi-0.5) = b2(xi+1)2 + a2(yi-0.5)2 - a2b2

若 di < 0，表示像素点 P1 和 P2 的中点在椭圆内，这时可取 P1 为下一个像素点。此时 xi+1 = xi + 1，yi+1 = yi，代入判别式 di 得到 di+1：

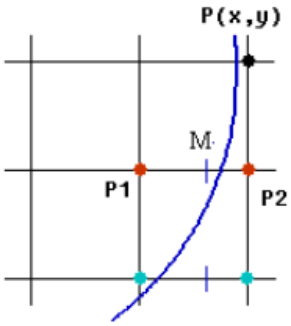
di+1 = F(xi+1+1, yi+1-0.5) = b2(xi+2)2 + a2(yi-0.5)2 - a2b2 = di + b2(2xi + 3)

计算出 di 的增量是 b2(2xi + 3)。同理，若 di >= 0，表示像素点 P1 和 P2 的中点在椭圆外，这时应当取 P2 为下一个像素点。此时 xi+1 = xi + 1，yi+1 = yi - 1，代入判别式 di 得到 di+1：

di+1 = F(xi+1+1, yi+1-0.5) = b2(xi+2)2 + a2(yi-1.5)2 - a2b2 = d1 + b2(2xi+3) + a2(-2yi+2)

计算出 di 的增量是 b2(2xi+3)+a2(-2yi+2)。计算 di 的增量的目的是减少计算量，提高算法效率，每次判断一个点时，不必完整的计算判别式 di，只需在上一次计算出的判别式上增加一个增量即可。

接下来继续讨论下部区域椭圆弧的生成，如图（4）所示：



图（4）中点画椭圆算法对下部区域处理示意图

假设当前位置是 P(xi, yi)，则下一个可能的点就是 P 点左下方的 P1(xi -1, yi-1)点或下方的 P2(xi, yi-1)点，取舍的方法同样取决于判别式 di，di 的定义如下：

di = F(xi+0.5, yi-1) = b2(xi+0.5)2 + a2(yi-1)2 - a2b2

若 di < 0，表示像素点 P1 和 P2 的中点在椭圆内，这时可取 P2 为下一个像素点。此时 xi+1 = xi + 1，yi+1 = yi - 1，代入判别式 di 得到 di+1：

di+1 = F(xi+1+0.5, yi+1-1) = b2(xi+1.5)2 + a2(yi-2)2 - a2b2 = di + b2(2xi+2)+a2(-2yi+3)

计算出 di 的增量是 b2(2xi+2)+a2(-2yi+3)。同理，若 di >= 0，表示像素点 P1 和 P2 的中点在椭圆外，这时应当取 P1 为下一个像素点。此时 xi+1 = xi，yi+1 = yi - 1，代入判别式 di 得到 di+1：

di+1 = F(xi+1+0.5, yi+1-1) = b2(xi+0.5)2 + a2(yi-2)2 - a2b2 = d1 + a2(-2yi+3)

计算出 di 的增量是 a2(-2yi+3)。

中点画椭圆算法从(0, b)点开始，第一个中点是(1, b - 0.5)，判别式 d 的初始值是：

d0 = F(1, b－0.5) = b2 + a2(-b+0.25)

上部区域生成算法的循环终止条件是：2b2(x + 1) >= 2a2(y - 0.5)，下部区域的循环终止条件是 y = 0，至此，中点画椭圆算法就可以完整给出。

1.1.5 六面体

六面体绘制是取鼠标点击点坐标，然后以此为中心绘制固定大小立方体，实现最终并没有实现消隐。

1.2 图形变换

1.2.1 平移

平移是通过获取用户输入平移向量，进行坐标变化实现。

1.2.2 旋转

旋转接受用户输入角度，将坐标利用公式进行变换，得到旋转后坐标。

1.2.3 缩放

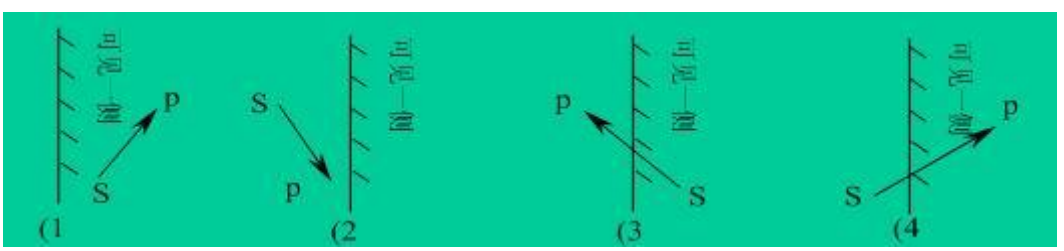
原理同上，进行新的坐标的获取，然后绘制。

1.2.4 裁剪

使用逐边裁剪算法（Sutherland-Hodgman 算法)，用户输入裁剪区域，将区域外的图像刷去，保留下来的部分即裁剪后部分。

基本思想：一次用窗口的一条边裁剪多边形。

考虑窗口的一条边以及延长线构成的裁剪线该线把平面分成两个部分:可见一侧；不可见一侧。多边形的各条边的两端点 S、P。它们与裁剪线的位置关系只有四种



情况（1）仅输出 1 个顶点 P；

情况（2）输出 0 个顶点；

情况（3）输出线段 SP 与裁剪线的 1 个交点 I；

情况（4）输出线段 SP 与裁剪线的 1 个交点 I 和 1 个终点 P

1.3 其他

1.3.1 填充

扫描填充算法

扫描线填充算法的基本思想是：用水平扫描线从上到下（或从下到上）扫描由多条首尾相连的线段构成的多边形，每根扫描线与多边形的某些边产生一系列交点。将这些交点按照 x 坐标排序，将排序后的点两两成对，作为线段的两个端点，以所填的颜色画水平直线。多边形被扫描完毕后，颜色填充也就完成了。扫描线填充算法也可以归纳为以下 4 个步骤：

（1）求交，计算扫描线与多边形的交点

（2）交点排序，对第 2 步得到的交点按照 x 值从小到大进行排序；

（3）颜色填充，对排序后的交点两两组成一个水平线段，以画线段的方式进行颜色填充；

（4）是否完成多边形扫描？如果是就结束算法，如果不是就改变扫描线，然后转第 1 步继续处理；

1.3.2 保存

未能实现

2 性能测试

2.1 图形绘制方面

直线，曲线，椭圆，圆形，多边形正确实现

六面体可以进行绘制，但消隐未能实现

2.2 图形变换方面

平移以及缩放针对所有图形均可正确实现

直线旋转可以实现，其他图形存在问题

多边形裁剪以及多边形填充正确实现

2.3 其他

保存功能未实现

3 参考资料

《计算机图形学教程》
DDA 以及 Bresenham 算法：http://blog.csdn.net/clever101/article/details/6076841
B 样条曲线绘制：http://blog.csdn.net/lafengxiaoyu/article/details/51291522
中心椭圆算法画图：http://blog.csdn.net/hw140701/article/details/53219005

图像旋转：http://blog.csdn.net/wonengguwozai/article/details/52049092
多边形裁剪：http://bbs.csdn.net/topics/340020792
http://blog.csdn.net/damotiansheng/article/details/43274183

图形填充：http://blog.csdn.net/xiaowei_cqu/article/details/7712451

致谢 在此,向对本文的工作给予支持和建议的同学表示感谢。