计算机图形学实验一：直线的Bresenham算法和DDA算法实现

原创

孜迹 于 2019-10-20 10:00:01 发布 3082 收藏 39

文章标签： 计算机图形学 直线的DDA算法 直线的Bresenham算法 坐标放大20倍

本文链接：https://blog.csdn.net/weixin_42752761/article/details/102646175

计算机图形学实验一：直线的Bresenham算法和DDA算法实现

解救众多被计算机图形学实验所困扰的学生党们，本博客仅粘贴代码，直线的Bresenham算法和DDA算法的原理请自行百度或Google，网上从来不缺原理。

ps：本代码属于硬核坐标放大，改进版坐标放大的方法请看博主另一篇博客计算机图形学实验三：多边形填充算法之活性边表方法实现中的坐标放大方法，比这里面的放大方法好了不下100倍！只是博主做完实验一身轻松，就不想再倒回去折腾了，你们就自己动手改吧，不然就用我这个硬核坐标放大也是没有问题的，就是解释器爱稍微麻烦了一点。

代码在此，参考为上，借鉴最佳

```
#include "pch.h"
#include <iostream>
#include<GL/glut.h>
#include<math.h>
#include<Windows.h>

const int WindowWidth = 800, WindowHeight = 800;
//DDA算法
void DDALine(int x0, int y0, int x1, int y1)
{
 int x01 = x0 * 20, y01 = y0 * 20, x11 = x1 * 20, y11 = y1 * 20;
 int dx = x11 - x01, dy = y11 - y01;
 if (dx != 0)
 {
  int k = dy * 20 / dx;
  if ((k > 20)||(k<-20)) {
   if (dy < 0) {
    int tempx0 = x01, tempy0 = y01;
    x01 = x11, y01 = y11;
    x11 = tempx0, y11 = tempy0;
   }
   int x = x01;
   k = dx * 20 / dy;
   int tempx = 0;
   glVertex2i(x01, y01);
   for (int y = y01 + 20; y <= y11;)
   {
    x = x + k;
    if (x < 0) {
     if (x % 20 == -10)
     {
      tempx = x / 20;
      glVertex2i(tempx * 20, y);
```

```
    }
    else
     tempx = (x - 10) / 20;
   }
   else
    tempx = (x + 10) / 20;
   glVertex2i(tempx * 20, y);
   y = y + 20;
  }
 }
 else{
  if (dx < 0) {
   int tempx0 = x01, tempy0 = y01;
   x01 = x11, y01 = y11;
   x11 = tempx0, y11 = tempy0;
  }
  int y = y01;
  int tempy = 0;
  glVertex2i(x01, y01);
  for (int x = x01 + 20; x <= x11;)
  {
   y = y + k;
   if (y < 0) {
    if (y % 20 == -10) {
     tempy = y / 20;
     glVertex2i(x, tempy * 20);
    }
    else
     tempy = (y - 10) / 20;
   }
   else
    tempy = (y + 10) / 20;
   glVertex2i(x, tempy * 20);
   x = x + 20;
  }
 }
}
else
{
 if (dy < 0) {
  int tempx0 = x01, tempy0 = y01;
  x01 = x11, y01 = y11;
  x11 = tempx0, y11 = tempy0;
 }
 for (int y = y01; y <= y11;) {
  glVertex2i(x01, y);
  y += 20;
 }
}
}

//Bresenham算法实现
void BresenhamLine(int x0, int y0, int x1, int y1) {
 int x01 = x0 * 20, y01 = y0 * 20, x11 = x1 * 20, y11 = y1 * 20;
 int dx = x11 - x01, dy = y11 - y01;
 if (dx == 0) {
  if (dy < 0) {
   int tempx0 = x01, tempy0 = y01;
   x01 = x11, y01 = y11;
```

```
  x11 = tempx0, y11 = tempy0;
 }
 for (int y = y01; y <= y11;) {
  glVertex2i(x01, y);
  y += 20;
 }
}
else {
 if (((dy > dx) && (dx < 0) && (dy * dx > 0)) || ((dy < dx) && (dy > 0))) {  //0<k<1
  int tempy = y01;
  int e = -dx;
  if (dx < 0) {
   int tempx0 = x01, tempy0 = y01;
   x01 = x11, y01 = y11;
   x11 = tempx0, y11 = tempy0;
   dx = x11 - x01, dy = y11 - y01;
   e = -dx;
   tempy = y01;
  }
  for (int x = x01; x <= x11;) {
   if (e < 0) {
    glVertex2i(x, tempy);
    e = e + 2 * dy;
    x = x + 20;
   }
   else
   {
    e = e - 2 * dx;
    tempy = tempy + 20;
    glVertex2i(x, tempy);
    e = e + 2 * dy;
    x = x + 20;
   }
  }
 }
 else if (((dy < dx) && (dx < 0)) || ((dy > dx) && (dx > 0))) {  //k>1
  int tempx = x01;
  int e = -dy;
  if (dx < 0) {
   int tempx0 = x01, tempy0 = y01;
   x01 = x11, y01 = y11;
   x11 = tempx0, y11 = tempy0;
   dx = x11 - x01, dy = y11 - y01;
   e = -dy;
   tempx = x01;
  }
  for (int y = y01; y <= y11;) {
   if (e < 0) {
    glVertex2i(tempx, y);
    e = e + 2 * dx;
    y = y + 20;
   }
   else
   {
    e = e - 2 * dy;
    tempx = tempx + 20;
    glVertex2i(tempx, y);
    e = e + 2 * dx;
    y = y + 20;
   }
```

```
  }
 }
 else if (((dy + dx < 0) && (dx > 0)) || ((dy + dx > 0) && (dx < 0))) {  //k<-1
  int tempx = x01;
  int e = -dy;
  if (dy < 0) {
   int tempx0 = x01, tempy0 = y01;
   x01 = x11, y01 = y11;
   x11 = tempx0, y11 = tempy0;
   dx = x11 - x01, dy = y11 - y01;
   e = -dy;
   tempx = x01;
  }
  for (int y = y01; y <= y11;) {
   if (e < 0) {
    glVertex2i(tempx, y);
    e = e - 2 * dx;
    y = y + 20;
   }
   else
   {
    e = e - 2 * dy;
    tempx = tempx - 20;
    glVertex2i(tempx, y);
    e = e - 2 * dx;
    y = y + 20;
   }
  }
 }
 else {     //-1 < k <0
  int tempy = y01;
  int e = -dx;
  if (dx < 0) {
   int tempx0 = x01, tempy0 = y01;
   x01 = x11, y01 = y11;
   x11 = tempx0, y11 = tempy0;
   dx = x11 - x01, dy = y11 - y01;
   e = -dx;
   tempy = y01;
  }
  for (int x = x01; x <= x11;) {
   if (e < 0) {
    glVertex2i(x, tempy);
    e = e - 2 * dy;
    x = x + 20;
   }
   else
   {
    e = e - 2 * dx;
    tempy = tempy - 20;
    glVertex2i(x, tempy);
    e = e - 2 * dy;
    x = x + 20;
   }
  }
 }
}
}
```

```
void resetSize(int w, int h)
{
 glutReshapeWindow(WindowWidth, WindowHeight);
}

void inIt()
{
 glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
 glutInitWindowPosition(50, 50);
 glutInitWindowSize(WindowWidth, WindowHeight);
 glutCreateWindow("Line Demo");
 gluOrtho2D(-WindowWidth / 2, WindowWidth / 2, -WindowHeight / 2, WindowHeight / 2);
 glutReshapeFunc(resetSize);

 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(0.1, 0.1, 0.1);
 glBegin(GL_POINTS);
 for (int x = (-WindowWidth); x < WindowWidth; x++) {
  for (int y = (-WindowHeight); y < WindowHeight; y += 20) {
   glVertex2i(x, y);
  }
 }
 for (int y = (-WindowHeight); y < WindowHeight; y++) {
  for (int x = (-WindowWidth); x < WindowWidth; x += 20) {
   glVertex2i(x, y);
  }
 }
 glEnd();
}

void Draw()
{
 glBegin(GL_POINTS);
 //set color as white
 glColor3f(10, 10, 10);
 glVertex2i(0, 0);
 //DDALine(-10, -3, 5, 17); //k>1   ok
 //DDALine(-10, -3, 10, 12);  //0<k<1    ok
 //DDALine(-10, -3, 10, -18);  //-1<k<0
 //DDALine(-2, 4, 2, -2);    //k<-1   ok
 //DDALine(-2, 2, 2, -2);    //k = 1 \ -1  ok
 //DDALine(-2, 4, 2, 4);    //k = 0 \ none  ok

 //BresenhamLine(-10, -3, 5, 17);  //k>1   ok
 //BresenhamLine(9, 2, -3, 10);  //0<k<1    ok
 //BresenhamLine(-10, -3, 10, -18);  //-1<k<0   ok
 //BresenhamLine(-2, 4, 2, -2);    //k<-1   ok
 //BresenhamLine(-2, 2, 2, -2);    //k = 1 \ -1  ok
 //BresenhamLine(-2, 4, -2, -4);    //k = 0 \ none  ok
 glEnd();
 glFlush();
}

 int main(int argc, char** argv) {
  glutInit(&argc, argv);
  inIt();
  glutDisplayFunc(Draw);
  glutMainLoop();
  return 0;
```

```
}
```

pch.h文件源码

```
#ifndef PCH_H
#define PCH_H

// TODO: 添加要在此处预编译的标头

#endif //PCH_H
```

还是那句话，参考为上，借鉴更佳，共勉。