**模块1 扫描转换算法**

**一 实验目的**

1. 编写直线、弧线的光栅扫描转换算法，并对线宽与线形的算法加以探讨
2. 用DDA算法、中点画线算法、Bresenham画线算法绘制直线（如果键盘输入数据，给出数据值；如果绘制图案，图案中应包含各种斜率；如果鼠标确定任意两点，给出操作说明）
3. 意识到各类直线的光栅扫描转换算法的局限性，并尝试改进后实现任意斜率的直线的画法

**二 实验理论分析**

**DDA法：**

1. 主要思想：主要采用增量算法思想，即当x每递增1时，y会递增k。而由于像素只能取整数，所以对(y+0.5)取整后标注该像素。
2. 局限性：当k的绝对值小于1时，能够较好的描绘直线；但是当k的绝对值大于1时，光栅点分布较为稀疏，像素标注情况较差。
3. 改进方法：当k的绝对值大于1时，将x和y的位置互换。即取x和y变化大的轴作为参考轴，来保证直线被光栅化后有足够多的像素。

**中点画线法：**

1. 主要思想：采用直线的一般式方程，即ax+by+c=0。构造判别式为d=F(xi+1,yi+0.5)=a(xi+1)+b(yi+0.5)+c，当d＜0时，取y+1，当d≥0时，取y。
2. 局限性：计算量太太大，不如DDA算法；只能绘制斜率在0到1之间的正斜率直线。
3. 改进方法：构造判别式为d0=F(x0+1,y0+0.5)=a+0.5b，重构在d的正负影响下dnew和dold之间的关系，并优化d为整数运算。根据dx和dy的大小关系，以跨步大的轴为步长方向，另一个轴为d的迭代方向。

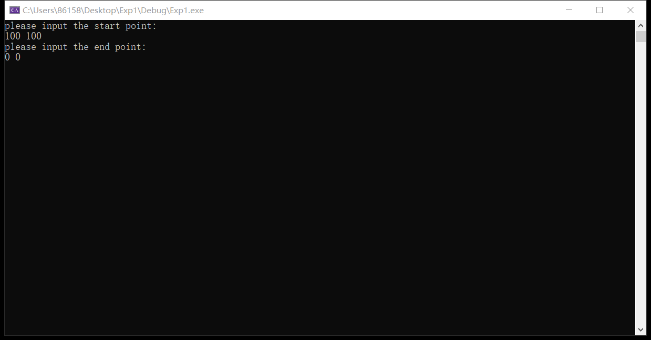
**Bresenham法：**

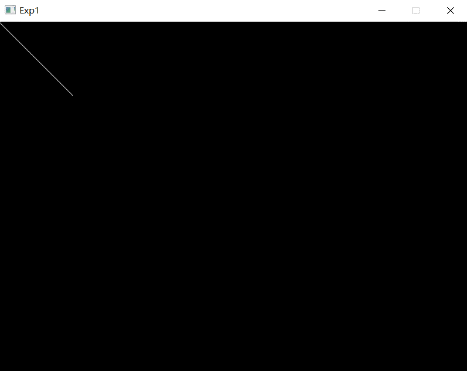
1. 主要思想：通过计算像素点之间的差值来决定下一个像素点的位置。一旦d≥1，就把它减去1，保证d在0、1之间，即始终有d∈[0,1)。
2. 局限性：计算不方便；只能绘制斜率在0到1之间的正斜率直线。
3. 改进方法：对于计算方便而言，第一种改进方法是令e=d-0.5，逐步迭代后判断e的正负并更改e；第二种改进方法是用e\*2△x替换e，从而获得整数算法并避免除法。对于任意斜率而言，可以根据dx和dy的大小关系，以跨步大的轴为步长方向，另一个轴为e的迭代方向。

**三 实验内容**

1：用DDA算法绘制任意斜率的直线

实验结果如下图所示：

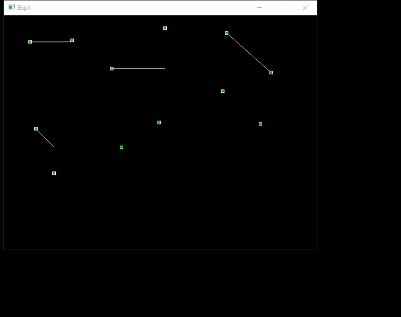
第一步：输入起点的坐标和终点的坐标（【100，100】为起点，【0，0】为终点）

第二步：程序自动连接上述两点

2：用中点画线和Bresenham画线算法绘制部分斜率的直线

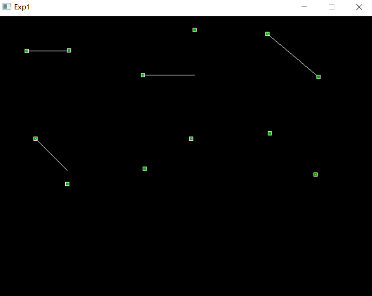
**2.1-Bresenham**画线算法实验结果如下图所示：

第一行的操作分别为：水平从左向右画线、从左下向右上画线、从左上向右下画线（不超过45度）；

第二行的操作分别为：从左上向右下画线（超过45度）、从右上向左下画线、从右下向左上画线；

**2.2-中点画线**算法实验结果如下图所示：

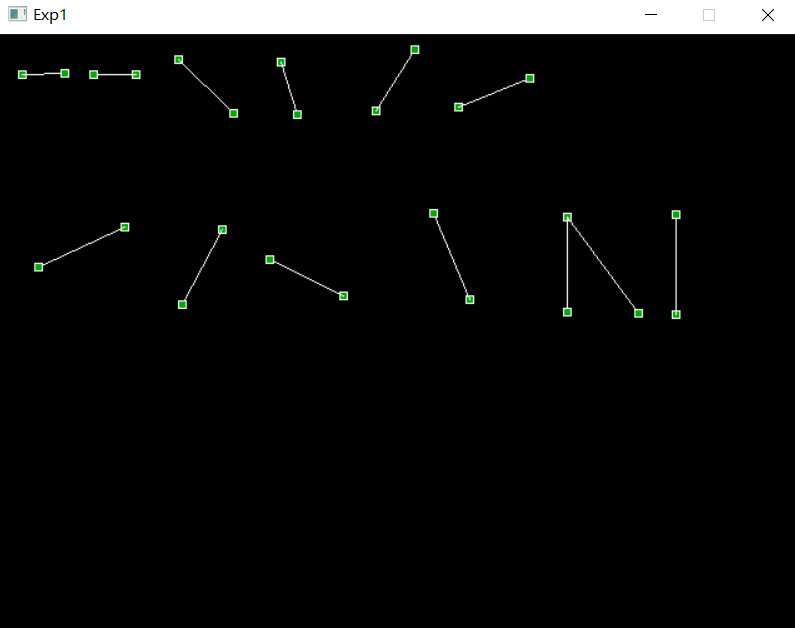
第一行的操作分别为：水平从左向右画线、从左下向右上画线、从左上向右下画线（不超过45度）；

第二行的操作分别为：从左上向右下画线（超过45度）、从右上向左下画线、从右下向左上画线；

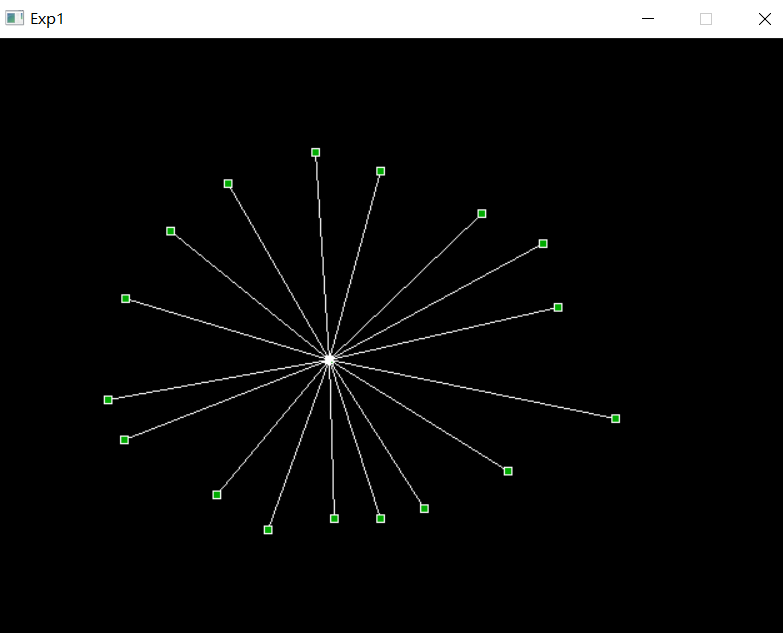
3：用中点画线和Bresenham画线算法绘制任意斜率的直线

**3.1-Bresenham**画线算法实验结果如下图所示：

第一行的操作分别为：水平方向画线（左->右，右->左）、从左上向右下画线（k的绝对值大于1，k的绝对值小于1）、从左下向右上画线（k的绝对值大于1，k的绝对值小于1）；

第二行的操作分别为：从右上向左下画线（k的绝对值大于1，k的绝对值小于1）、从右下向左上画线（k的绝对值大于1，k的绝对值小于1）、竖直方向画线（上->下，下->上）；

**3.2-中点画线**算法实验结果如下图所示：

从任一点出发，即当前点为起点，在四周画出其他的任意斜率的直线。最终结果为如下的辐射状的图形，即可证明该算法可以画出任意斜率的直线。

**四 程序说明**

Project中程序的调用：

将当前cpp文件的属性——常规——从生成中排除中选择否，其他文件选择是，即可运行当前的cpp文件

|  |
| --- |
| 1题 |
| //////////////////////////////////////////////////////  // 程序名称：DDA  // 功 能：用DDA算法绘制任意斜率的曲线  // 编译环境：VS2019，EasyX\_20220116  // 作 者：夏婉可<2020301010225><1597493790@qq.com>  // 最后修改：2022-3-3  #include <graphics.h>  #include <math.h>  #include <conio.h>  #include <iostream>  using namespace std;  void DDA\_Line(int x0, int y0, int x1, int y1, int color){  int dx = x1 - x0, dy = y1 - y0, step, k;  float x = x0, y = y0, xIncre, yIncre;  //斜率绝对值大于1的时候，以 y 的变化为基准  if (abs(dx) > abs(dy)) {  step = abs(dx);  }  else {  step = abs(dy);  }  //x和y的增量计算  xIncre = (float)dx / (float)step;  yIncre = (float)dy / (float)step;  for (k = 0; k < step; k++) {  putpixel(int(x + 0.5f), (int)(y + 0.5f), color);  x += xIncre;  y += yIncre;  }  }  int main(){  int x1, y1, x2, y2;  cout << "please input the start point:" << endl;  cin >> x1 >> y1;  cout << "please input the end point:" << endl;  cin >> x2 >> y2;    initgraph(640, 480);  DDA\_Line(x1, y1, x2, y2, WHITE);  \_getch();  closegraph();  return 0;  } |
| 2题：中点画线 |
| //////////////////////////////////////////////////////  // 程序名称：中点画线  // 功 能：用中点画线算法绘制部分斜率的曲线  // 编译环境：VS2019，EasyX\_20220116  // 作 者：夏婉可<2020301010225><1597493790@qq.com>  // 最后修改：2022-3-3  #include <graphics.h>  #include <math.h>  #include <conio.h>  #include <iostream>  using namespace std;  void MidPointLine(int x0, int y0, int x1, int y1, int color) {  int dx, dy, increE, increNE, d, x, y;  dx = x1 - x0;  dy = y1 - y0;  d = dx - 2 \* dy;  increE = -2 \* dy;  increNE = 2 \* (dx - dy);  x = x0;  y = y0;  putpixel(x, y, color);  while (x < x1) {  if (d > 0) {  d += increE;  x++;  }  else {  d += increNE;  y++;  x++;  }  putpixel(x, y, color);  }  }  int main(){  initgraph(640, 480);  ExMessage m;  int x0, y0, x1, y1;  while (true) {  m = getmessage(EX\_MOUSE | EX\_KEY);  switch (m.message) {  case WM\_LBUTTONDOWN:  x0 = m.x;  y0 = m.y;  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_RBUTTONDOWN:  x1 = m.x;  y1 = m.y;  MidPointLine(x0, y0, x1, y1, WHITE);  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_KEYDOWN:  if (m.vkcode == VK\_ESCAPE)  return 0; // 按 ESC 键退出程序  }  }  closegraph();  return 0;  } |
| 2题：Bresenham画线 |
| //////////////////////////////////////////////////////  // 程序名称：bresenham画线  // 功 能：用bresenham画线算法绘制部分斜率的曲线  // 编译环境：VS2019，EasyX\_20220116  // 作 者：夏婉可<2020301010225><1597493790@qq.com>  // 最后修改：2022-3-3  #include <graphics.h>  #include <math.h>  #include <conio.h>  #include <iostream>  using namespace std;  void bresenham(int x0, int y0, int x1, int y1, int color) {  int x, y, dx, dy, i, e;  dx = x1 - x0;  dy = y1 - y0;  e = -dx;  x = x0;  y = y0;  for (i = 0; i <= dx; i++) {  putpixel(x, y, color);  x++;  e += 2 \* dy;  if (e >= 0) {  y++;  e -= 2 \* dx;  }  }  }  int main() {  initgraph(640, 480);  ExMessage m;  int x0, y0, x1, y1;  while (true) {  m = getmessage(EX\_MOUSE | EX\_KEY);  switch (m.message) {  case WM\_LBUTTONDOWN:  x0 = m.x;  y0 = m.y;  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_RBUTTONDOWN:  x1 = m.x;  y1 = m.y;  bresenham(x0, y0, x1, y1, WHITE);  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_KEYDOWN:  if (m.vkcode == VK\_ESCAPE)  return 0; // 按 ESC 键退出程序  }  }  closegraph();  return 0;  } |
| 3题：中点画线 |
| //////////////////////////////////////////////////////  // 程序名称：中点画线：加强版  // 功 能：用中点画线算法绘制任意斜率的曲线  // 编译环境：VS2019，EasyX\_20220116  // 作 者：夏婉可<2020301010225><1597493790@qq.com>  // 最后修改：2022-3-3  #include <graphics.h>  #include <math.h>  #include <conio.h>  #include <iostream>  using namespace std;  void MidPointLine(int x0, int y0, int x1, int y1, int color)  {  int a, b, d1, d2, d, x, y;  float m;  if (x1 < x0) {  d = x0;  x0 = x1;  x1 = d;  d = y0;  y0 = y1;  y1 = d;  }  a = y0 - y1; //y差值  b = x1 - x0; //x差值  if (b == 0) {  m = -1 \* a \* 100;  }  else {  m = (float) a / (x0 - x1);  }  x = x0;  y = y0;  putpixel(x, y, color);  //斜率在0~1  if (m >= 0 && m <= 1){  d = 2 \* a + b;  d1 = 2 \* a;  d2 = 2 \* (a + b);  while (x < x1) {  if (d <= 0) {  x++;  y++;  d += d2;  }  else {  x++;  d += d1;  }  putpixel(x, y, color);  }  }  //斜率在-1~0  else if (m <= 0 && m >= -1) {  d = 2 \* a - b;  d1 = 2 \* a - 2 \* b;  d2 = 2 \* a;  while (x < x1) {  if (d > 0) {  x++;  y--;  d += d1;  }  else {  x++;  d += d2;  }  putpixel(x, y, color);  }  }  //斜率在1~∞  else if (m > 1) {  d = a + 2 \* b; d1 = 2 \* (a + b), d2 = 2 \* b;  while (y < y1) {  if (d > 0) {  x++;  y++;  d += d1;  }  else {  y++;  d += d2;  }  putpixel(x, y, color);  }  }  //斜率在∞~-1  else {  d = a - 2 \* b; d1 = -2 \* b, d2 = 2 \* (a - b);  while (y > y1) {  if (d <= 0) {  x++;  y--;  d += d2;  }  else {  y--;  d += d1;  }  putpixel(x, y, color);  }  }  }  int main() {  initgraph(640, 480);  ExMessage m;  int x0, y0, x1, y1;  while (true) {  m = getmessage(EX\_MOUSE | EX\_KEY);  switch (m.message) {  case WM\_LBUTTONDOWN:  x0 = m.x;  y0 = m.y;  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_RBUTTONDOWN:  x1 = m.x;  y1 = m.y;  MidPointLine(x0, y0, x1, y1, WHITE);  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_KEYDOWN:  if (m.vkcode == VK\_ESCAPE)  return 0; // 按 ESC 键退出程序  }  }  closegraph();  return 0;  } |
| 3题：Bresenham画线 |
| //////////////////////////////////////////////////////  // 程序名称：bresenham画线：加强版  // 功 能：用bresenham画线算法绘制任意斜率的曲线  // 编译环境：VS2019，EasyX\_20220116  // 作 者：夏婉可<2020301010225><1597493790@qq.com>  // 最后修改：2022-3-3  #include <graphics.h>  #include <math.h>  #include <conio.h>  #include <iostream>  using namespace std;  void bresenham(int x0, int y0, int x1, int y1, int color) {  int dx = abs(x1 - x0), sx = x0 < x1 ? 1 : -1;  int dy = abs(y1 - y0), sy = y0 < y1 ? 1 : -1;  int erro = (dx > dy ? dx : -dy) / 2;  while (putpixel(x0, y0, color), x0 != x1 || y0 != y1) {  int e2 = erro;  if (e2 > -dx) {  erro -= dy;  x0 += sx;  }  if (e2 < dy) {  erro += dx;  y0 += sy;  }  }  }  int main() {  initgraph(640, 480);  ExMessage m;  int x0, y0, x1, y1;  while (true) {  m = getmessage(EX\_MOUSE | EX\_KEY);  switch (m.message) {  case WM\_LBUTTONDOWN:  x0 = m.x;  y0 = m.y;  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_RBUTTONDOWN:  x1 = m.x;  y1 = m.y;  bresenham(x0, y0, x1, y1, WHITE);  setfillcolor(GREEN);  fillrectangle(m.x - 3, m.y - 3, m.x + 3, m.y + 3);  case WM\_KEYDOWN:  if (m.vkcode == VK\_ESCAPE)  return 0; // 按 ESC 键退出程序  }  }  closegraph();  return 0;  } |