```cpp
/**
* Program: dda.cpp
*/
#include <GL/glut.h>
#include <GL/glu.h>
#include <cmath>

void init()
{
    glClearColor(0.0, 0.0, 0.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0, 500.0, 0.0, 500.0);
}

void generateByDDA(int x0, int y0, int x1, int y1)
{
    int iter, steps;
    float deltaX = x1 - x0;
    float deltaY = y1 - y0;
    float x, y;
    if (abs(deltaX) > abs(deltaY)) {
        steps = abs(deltaX);
    }
    else {
        steps = abs(deltaY);
    }

    float xIncrement = deltaX / (float) steps;
    float yIncrement = deltaY / (float) steps;
    x = x0;
    y = y0;

    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_POINTS);
        for (iter = 0; iter < steps; iter++) {
            glVertex2i(x, y);
            x = x + xIncrement;
            y = y + yIncrement;
        }
    glEnd();
}


void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    generateByDDA(100, 200, 400, 200);

    //DDA suffer the effects of aliasing - diagonal line
    generateByDDA(100, 200, 400, 400);
    glFlush();
}


int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(300, 200);
    glutCreateWindow("Line generation using DDA LGA");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```