2D Line Clipping Simulation
Using Sutherland-Cohen line clipping algorithm
Simulated using OpenGl API

Introduction

In computer graphics, before scan converting it is often a good practice to clip the primitives since the clipped version might be much smaller and hence save time and processing. Primitives are usually defined in the real world, and their mapping from the real to the integer domain of the display might result in the overflowing of the integer values resulting in unnecessary artifacts

Theory

On given a clipping window and an external point (Fig. 1), the point is said to be visible if $X_{min} <= x <= X_{max}$ and $Y_{min} <= y <= Y_{max}$

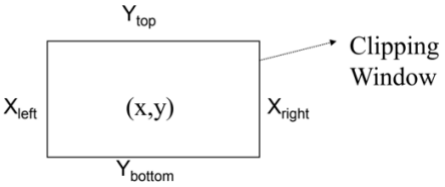

Fig. 1: Clipping a point

Clipping of all the points lying on a given line results in the clipped version of the line. But a point-by-point clipping results in an efficient algorithm.
Sutherland-Cohen Algorithm provides a more efficient method for clipping such lines.

Observations used in the formulation of the Sutherland-Cohen line clipping algorithm:

- If both the end points of a line are visible, then the line will be visible.
- For a given line, more than one invisible segments are possible but just one visible segment.
- The number of endpoints lying outside the window (1 or 2) is the number of intersections the line makes with the window.

Sutherland-Cohen line clipping algorithm-

1. Divide the screen into 9 regions and assign a bitcode[1] to each region (Fig. 2).

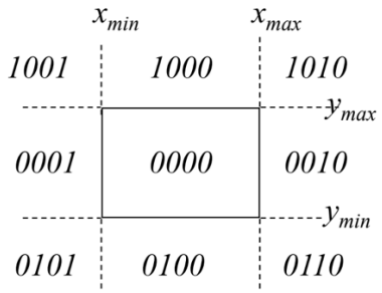

Fig. 1: 9 regions for quadrant 1

2. Find the bitcodes of the endpoints of the given line.
3. Test the outcode[2].
4. If the outcode is non-zero, the line is trivially rejected.
5. Find the edge that is intersecting with the clipping window and find the point of intersection.
6. Replace that endpoint with the point of intersection.
7. Repeat steps 2-6 for this newly obtained line.

---

1. Bitcode – It is a four bit binary code (bit-code) assigned to a point. Given point (x,y) and endpoints of the clipping window (X_{max}, Y_{max}, X_{min}, Y_{min}), the bitcode is decided as Bit1 → $y > Y_{max}$; Bit2 → $y < Y_{min}$; Bit3 → $x > X_{max}$; Bit4 → $x < X_{min}$.
2. Outcode – It is the logical AND of the four bits of the bitcodes of two points.

Implementation

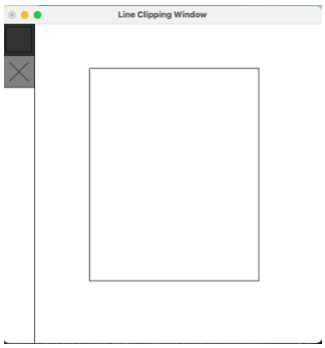

Fig. 3: Specifying clipping window

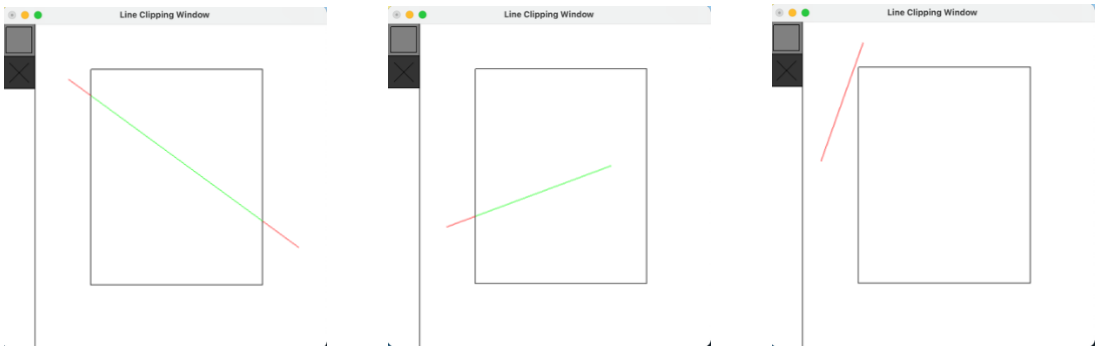

Fig. 4a,b,c: Getting a clipped line (--) by specifying the end points of a line (--).