CS770/870 Assignment 2

Due date: Monday, September 21st, 2020, before midnight.

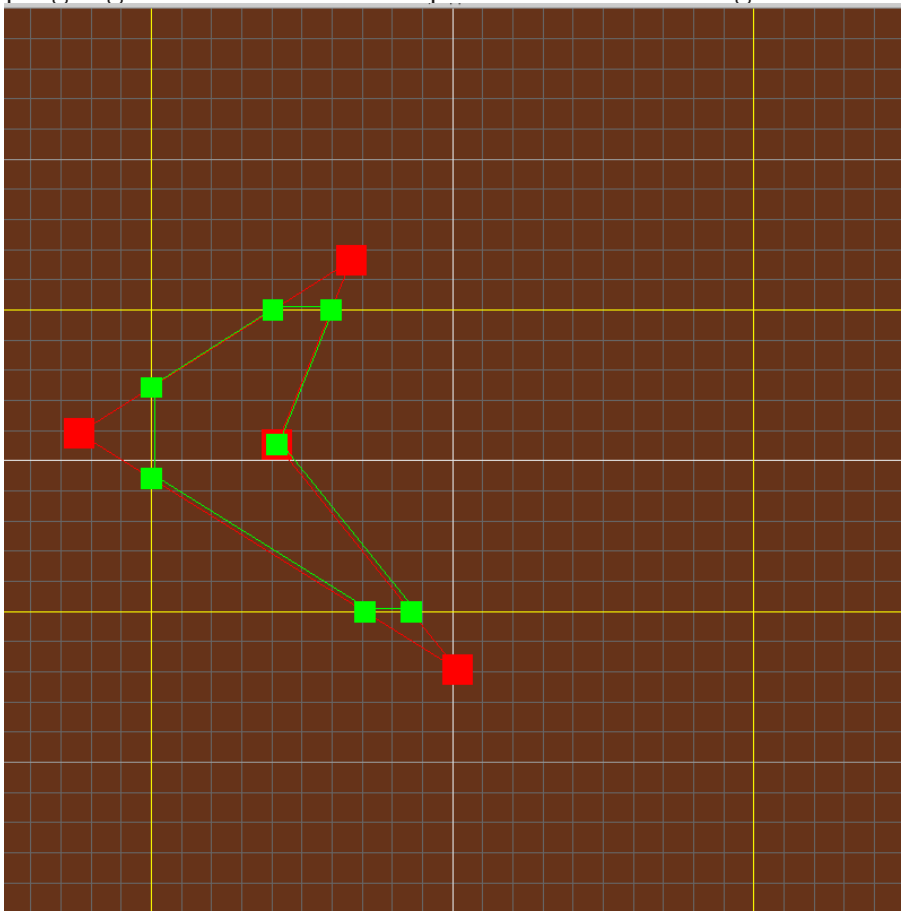
Lateness Penalty: Tue: 5% Wed: 10% Thu: 20% Fri: 50% Sat/Sun/Mon: 100%

Goals

In this assignment, you will gain more experience with the `GLFW` event model and 2D graphics. You will implement both the Sutherland-Hodgman polygon clipping algorithm, and a pixel-rasterizing algorithmm.

Tasks

1. [50 points] Write a program that demonstrates the Sutherland-Hodgman clipping algorithm. It should implement the following behavior:



.

   - There is a clipping window from x_left=-1 to x_right=+1, and y_bottom=-0.5 to y_top=+0.5. This window should be drawn with yellow lines that extend past the sides of the window.

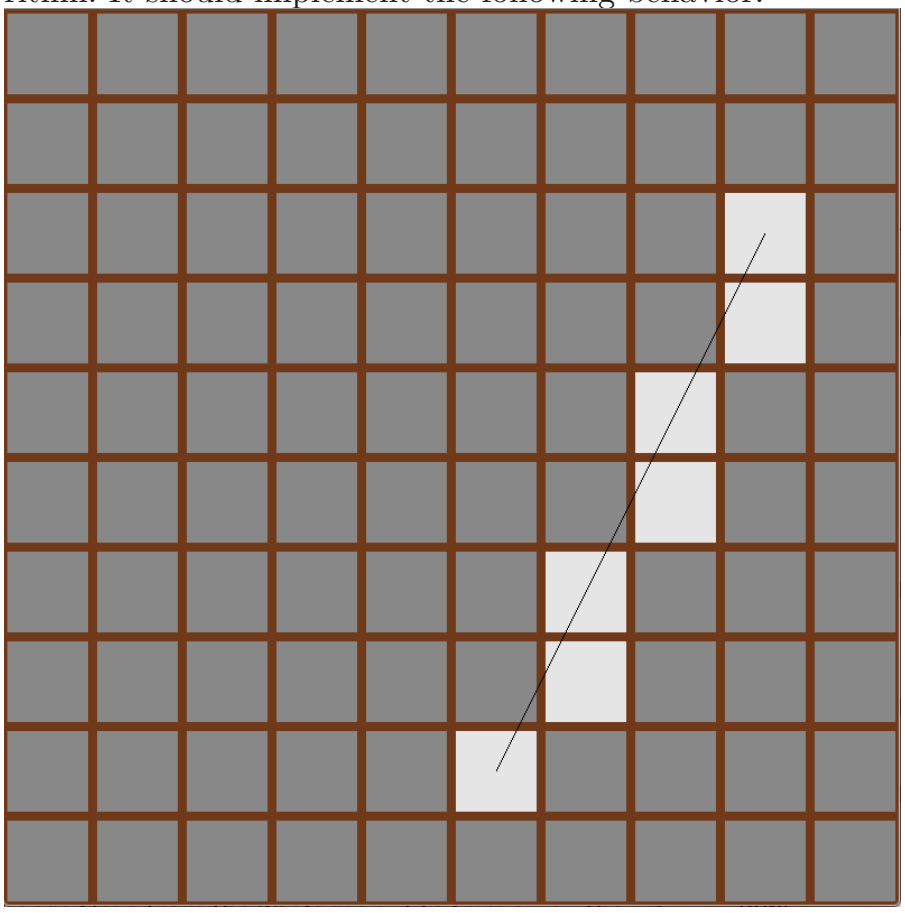
- Each click in an empty region will add another (x y) vertex to the current polygon. The polygon is stored as a sequence of (x y) points, and each click adds a point to the end of this sequence.

- Each polygon vertex is drawn as a 0.1×0.1 yellow square, and polygon edges are drawn as red line segments.

- A click on an existing vertex starts dragging it.

- Whenever the polygon is changed (by adding a new vertex, or dragging an existing one), a clipped polygon should be obtained. This polygon should be drawn with green lines and green 0.5×0.5 vertices.

2. [50 points] Implement a program that demonstrates a line rasterization algorithm. It should implement the following behavior:



.

- The window contains a 10×10 grid of grey squares, each one 0.9×0.9 .

- When the user clicks somewhere, and drags the mouse, the program will draw a black line segment that begins at the mouse-down pixel, and ends at the pixel where the mouse currently is.

- When the user releases the mouse, the line should remain, but should not be updated.

- The endpoints of the line segment should be at the centers of the corre-

sponding pixels.

    - Each pixel that the line is closest to should be drawn white, not gray.

New! Improved!

I'm supplying starting code. The code includes these changes:

- The `Mesh` class now has an `update_vertex` method, which lets you modify the coordinates of one vertex. You can now re-use `Mesh` objects, such as ones that are drawn with `GL_LINE_STRIP`s. You no longer have to re-allocate and delete `Mesh` objects.

- The `add_vertex` function (which adds an x y z point to an array of coordinates, with an associated element index), has been moved the `util.cpp` file.

Submitting Your Work

When you are done, go to `mycourses.unh.edu`, find the course, and the assignment. Then click `Submit`, and upload both `clipper.cpp` and `rasterizer.cpp`.