Cityscape Documented

I wanted to start off with a normal grid style manhatten type of city generation but was able to get a normal grid up and running very quickly. Randomized buildings, different variations based on heights etc. but I felt like it was too constricting and rigid so I opted for a more versatile city generation.

I looked into Voronoi Diagrams.
An almost perfect way to build a city but feels very rural however most cities are not always just grid shaped.

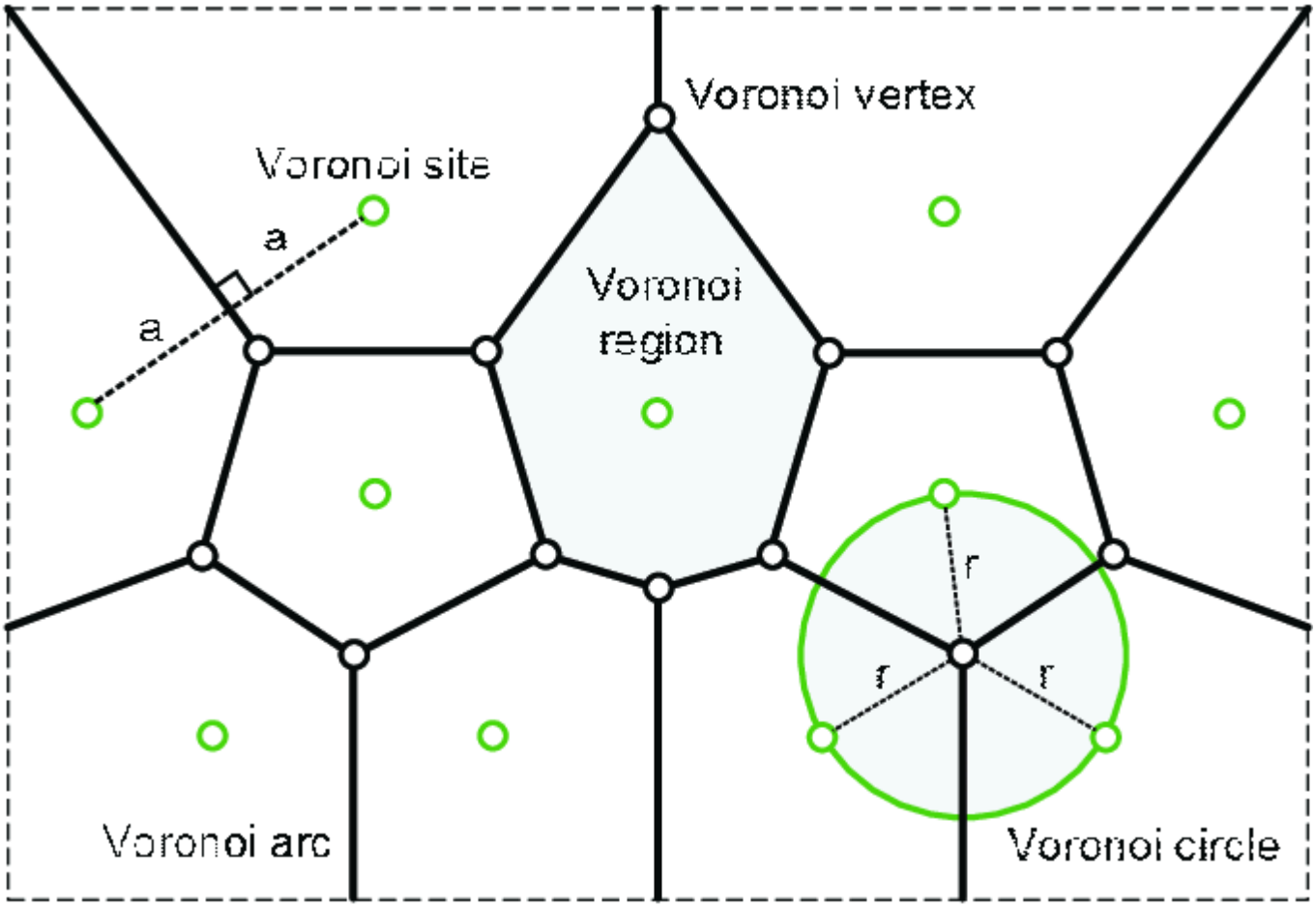
I tried making a voronoi diagram and have the roads placed THEN within the gaps have buildings placed. This proved impossible due to the requirement of having the buildings face the right direction as to not overlap the roads

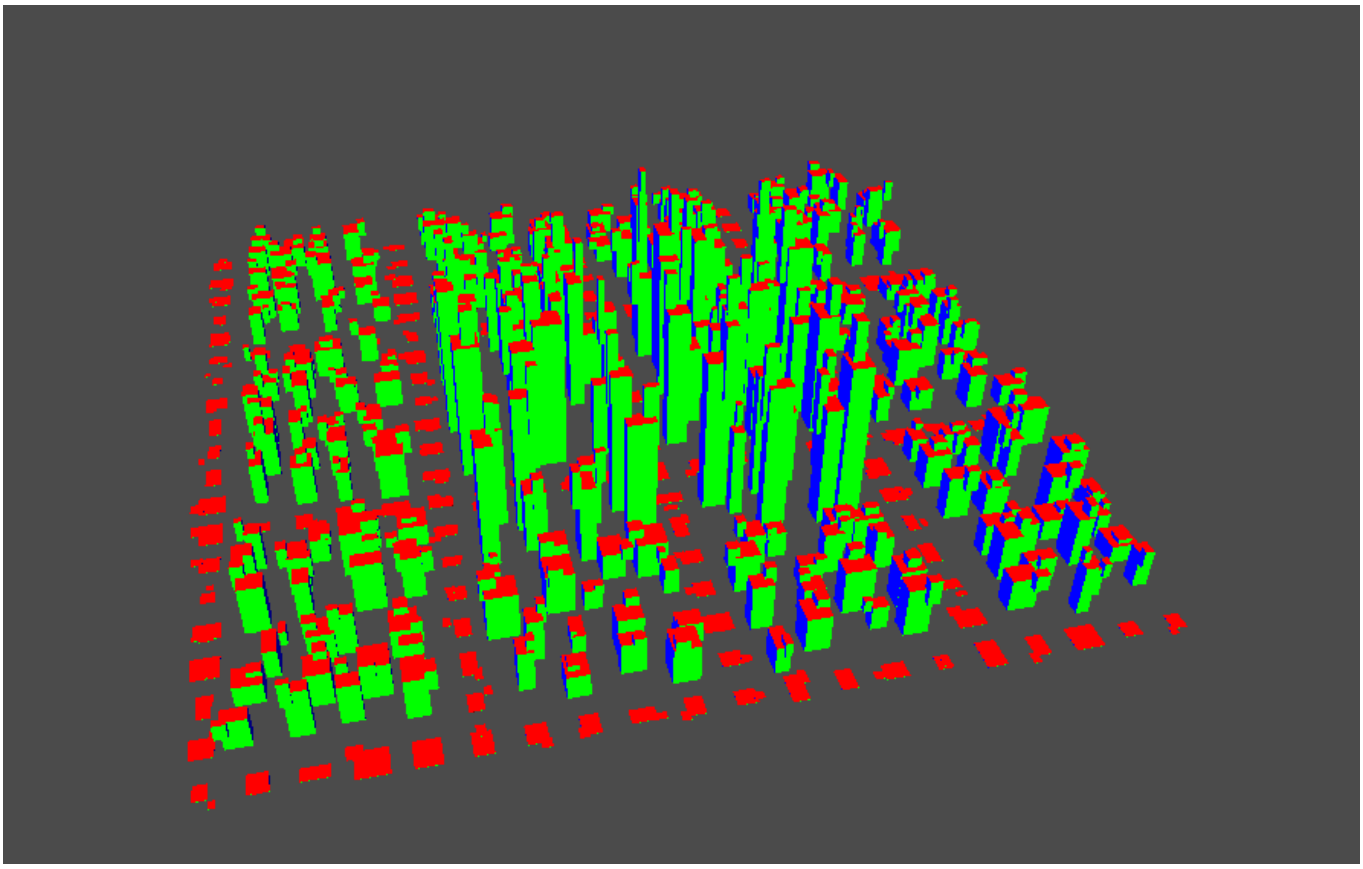
Manual voronoi diagram is very math intensive so I'll try going for Delaunay triangulation.

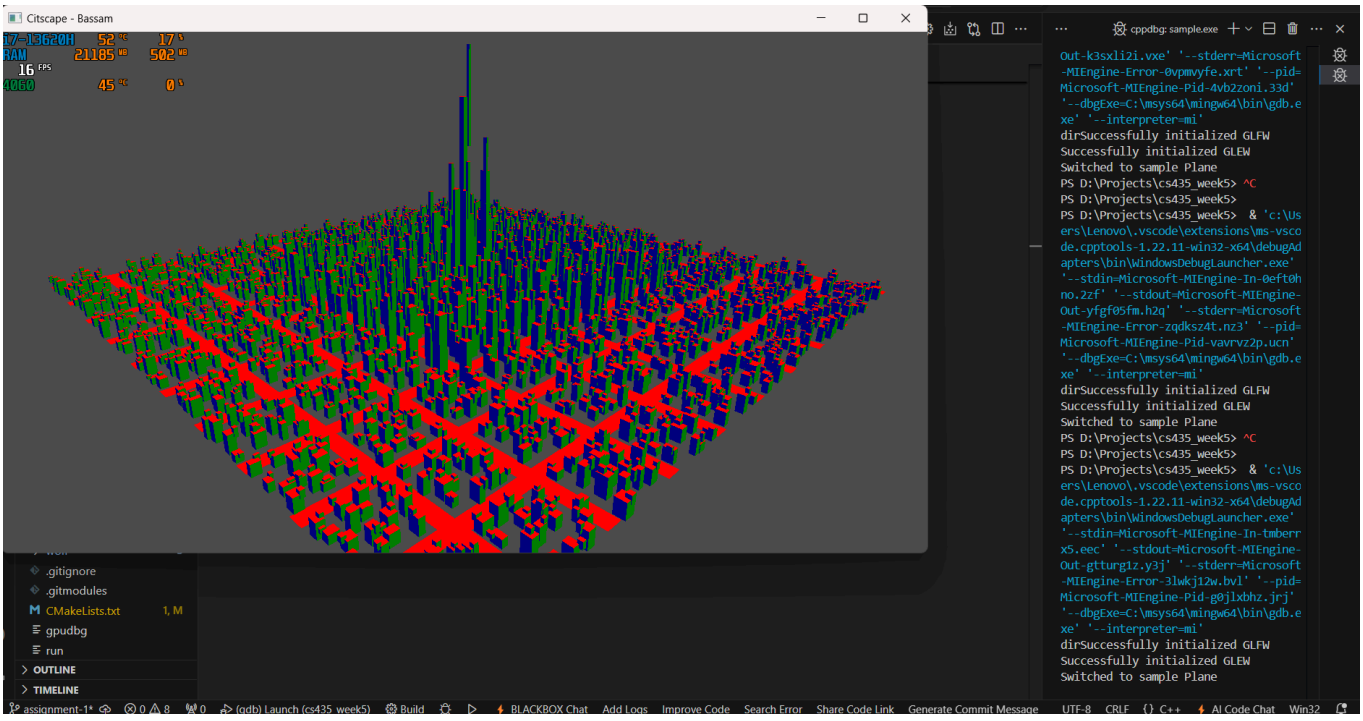
Texturing the road would become an issue moving forward as the city would be procedural, I'll worry about that later.

A Mathematical Guide to Social Distancing | Voronoi Diagrams
Voronoi Diagram by Allen Miu

This explained the idea pretty well, my thought process going forward is randomly adding sites on the ground.
Loop through each site
Then loop through each site again (that isnt the current site)
Find the perpendicularBisector of the two sites
I think I'll go for the point in the middle and the direction in perpendicular and draw that line.


Voronoi vertex
Voronoi site
a
a
Voronoi
region
r
r
r
Voronoi arc
Voronoi circle





My idea when it comes to cutting out perpendicular bisector from the voronoi cell's polygon is creating a line then finding the intersections within the line on the current polygon.

Clipping polygons mathematically seemed impossible until I discovered the Sutherland-Hodgman algorithm which is a simple way of clipping polygons.

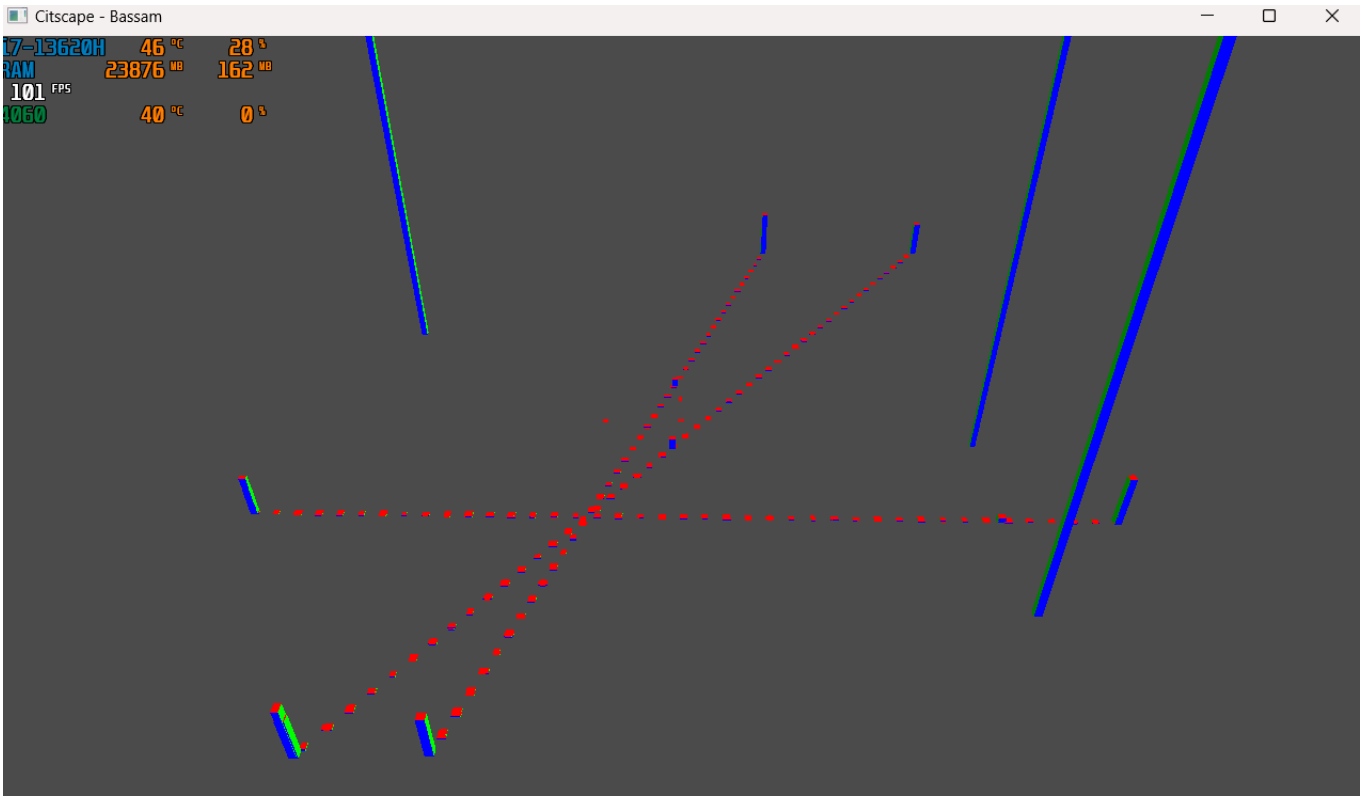
``There are four possible cases for any given edge of given polygon against current clipping edge e.

```
Both vertices are inside:
    Only the second vertex is added to the output list
First vertex is outside while second one is inside:
        Both the point of intersection of the edge with the clip boundary and
the second vertex are added to the output list
First vertex is inside while second one is outside:
    Only the point of intersection of the edge with the clip boundary is added
to the output list
Both vertices are outside:
    No vertices are added to the output list
```

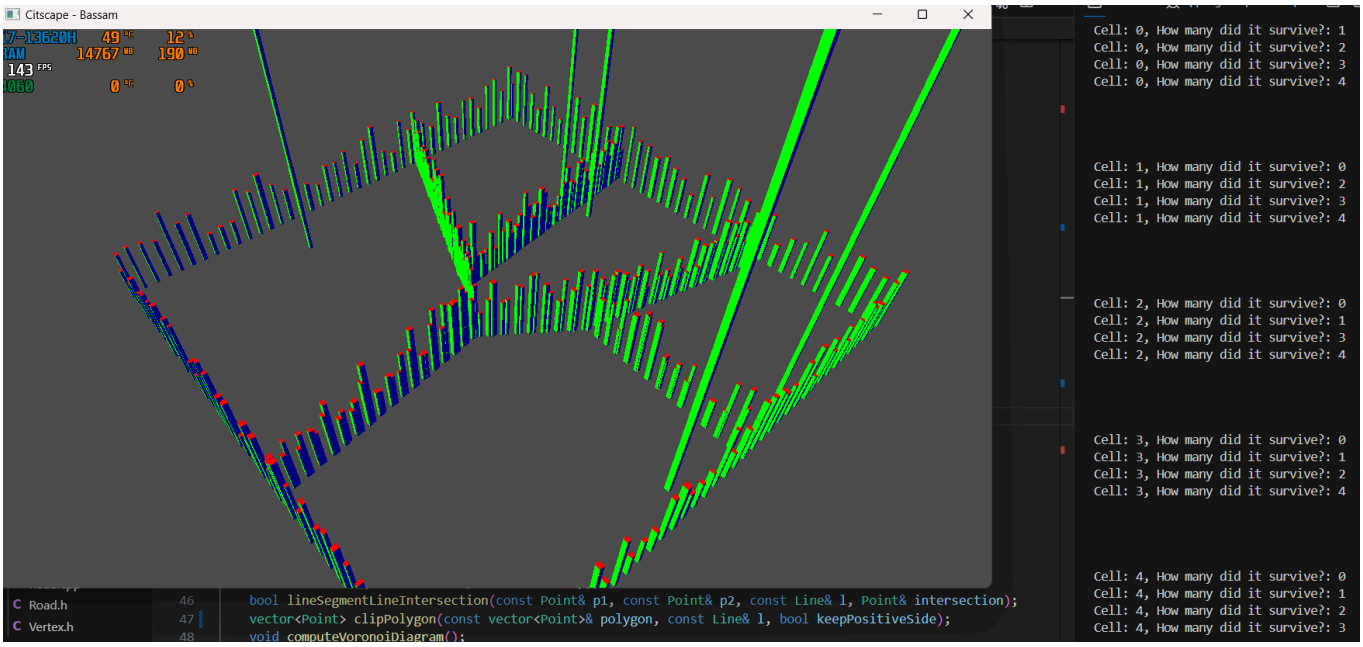
[Sutherland-Hodgeman Polygon Clipping Algorithm]
I want to modify this method to have it work on a line...
How it will work:

- Go through each vertex and its previous vertex (being last one in the beginning)
  Call them v0 and v1
- v0 outside line?

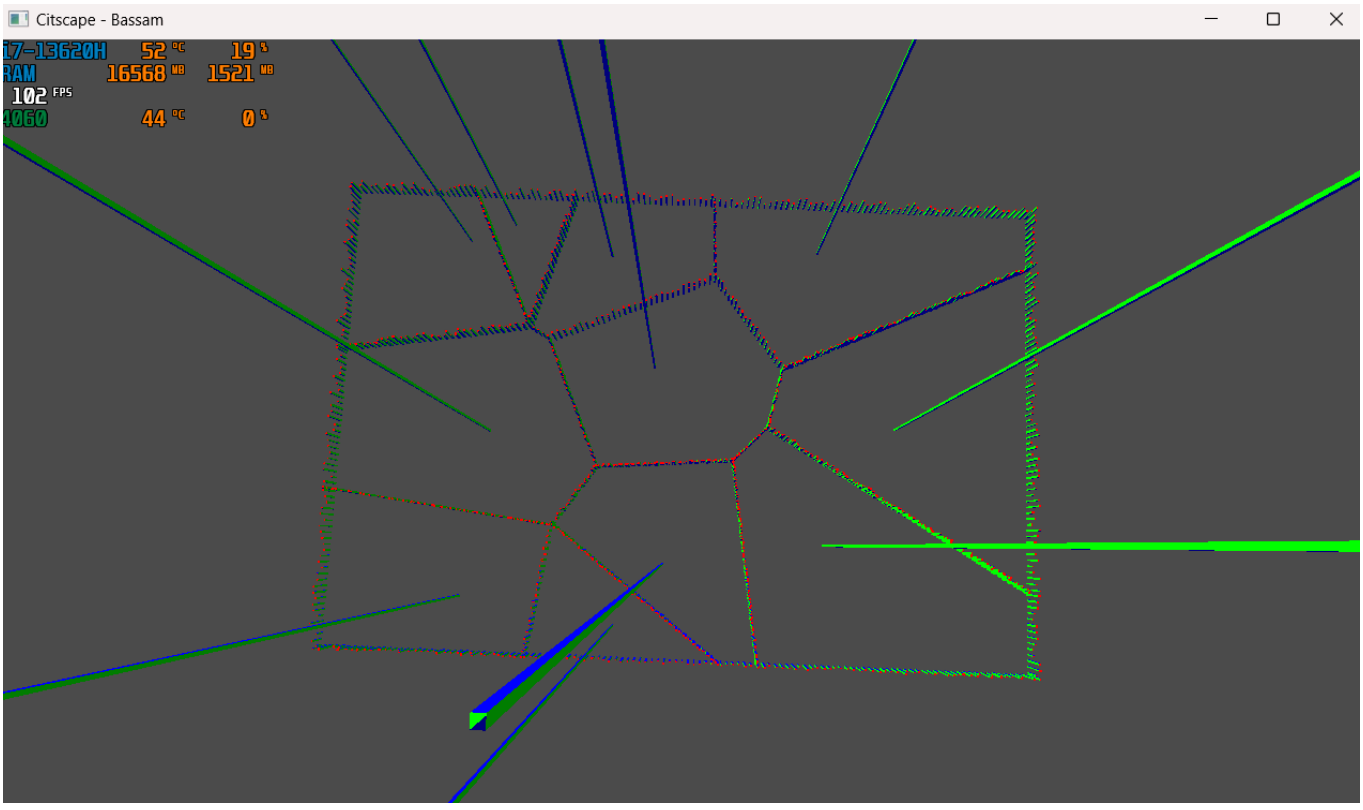
FINALLY GOT THIS SHIT TO WORK



Finally working

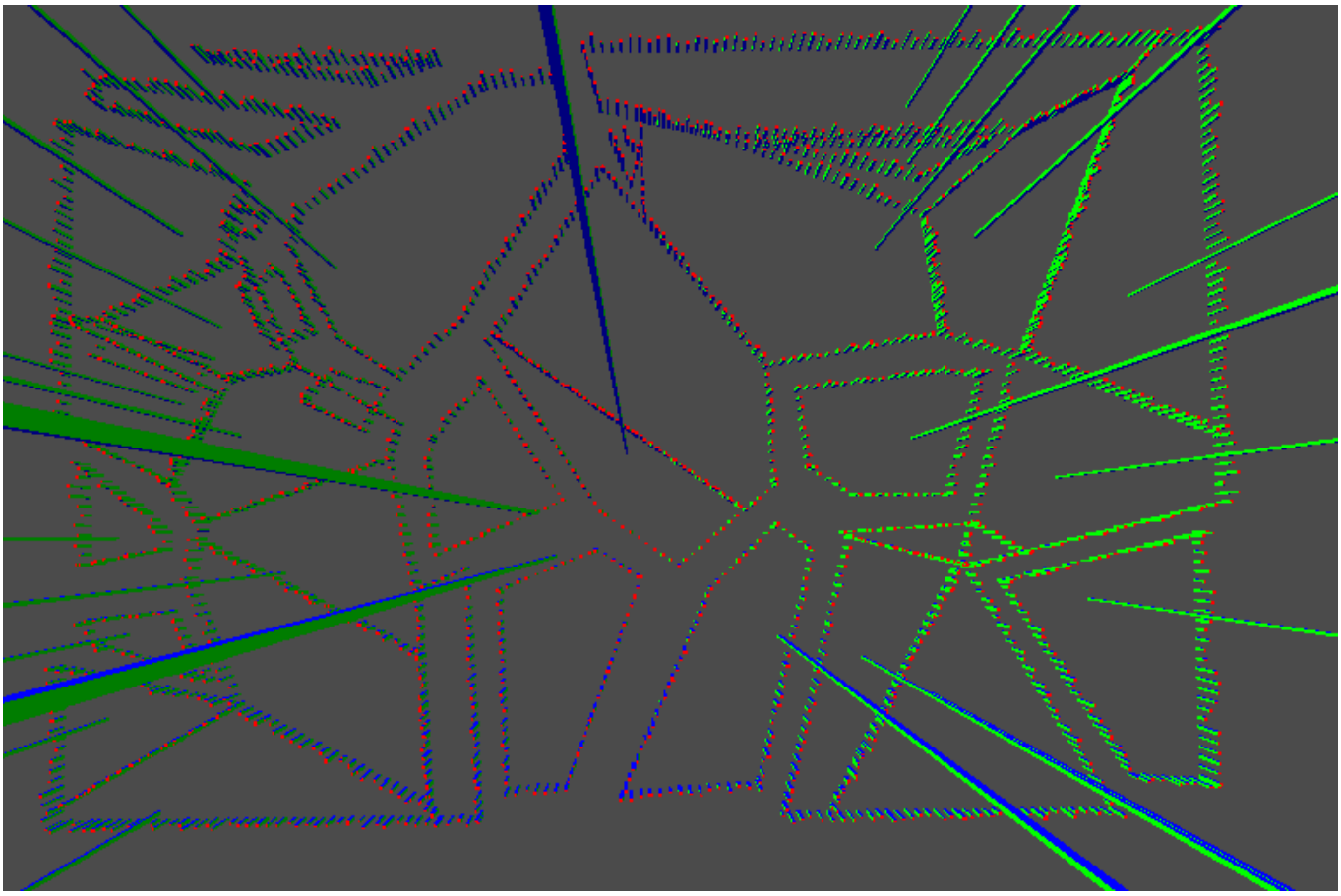


Took a while to get to this level

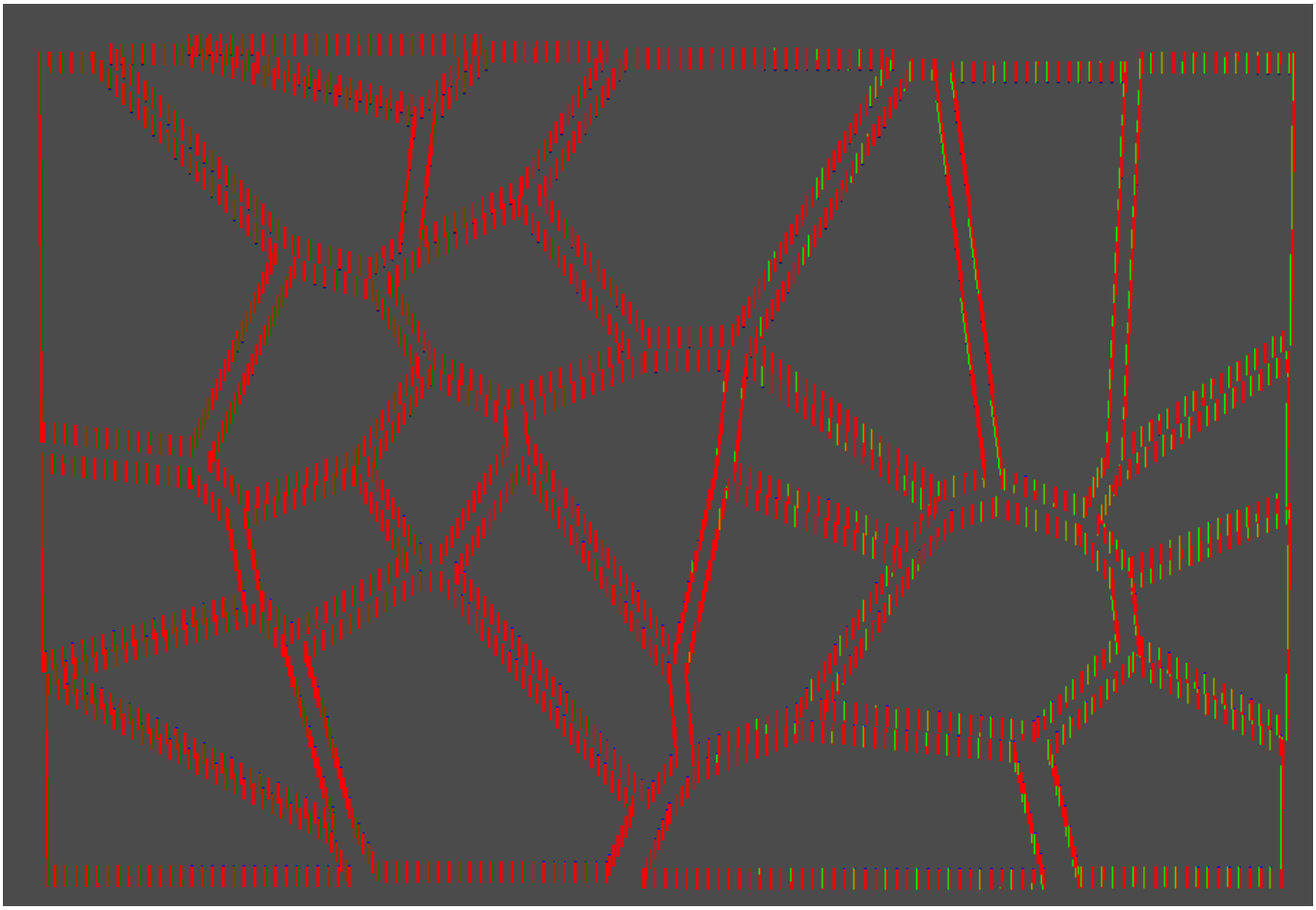


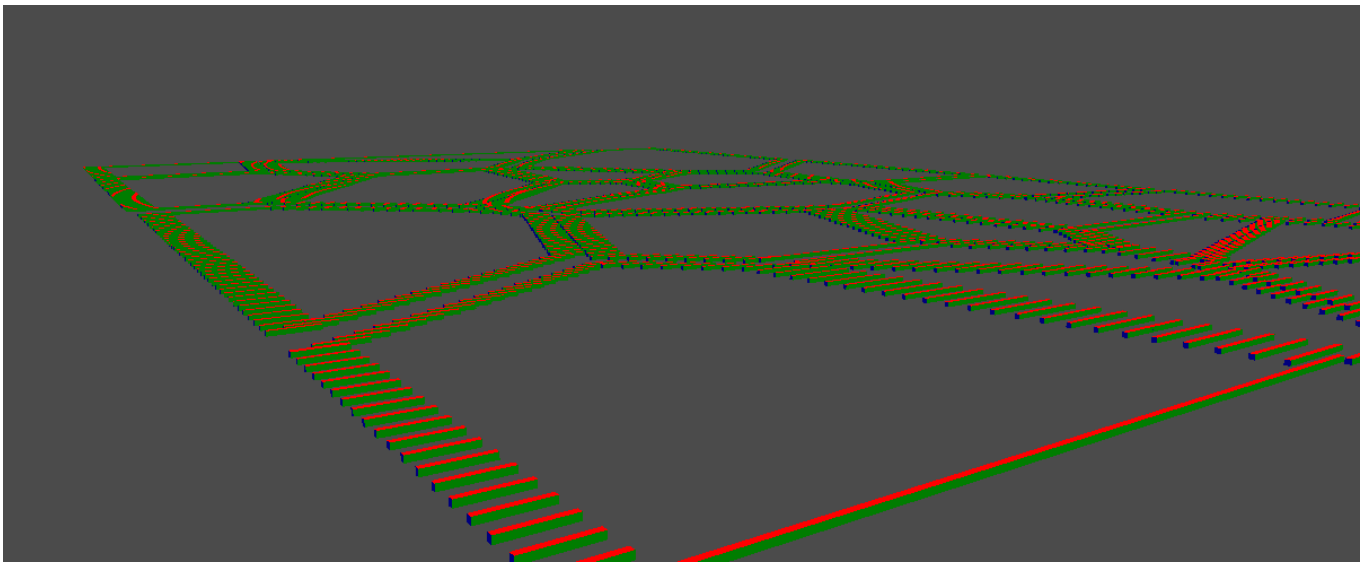
Next up:

- Subdivide big chunks into smaller chunks
  - Slicing across an edge and then again (perpendicularly)
- Scale chunks in to make room for roads
- Cut chunks up into a grid shape
- Scale chunks in (but less, side roads)
- Get those verticies and scale them up.



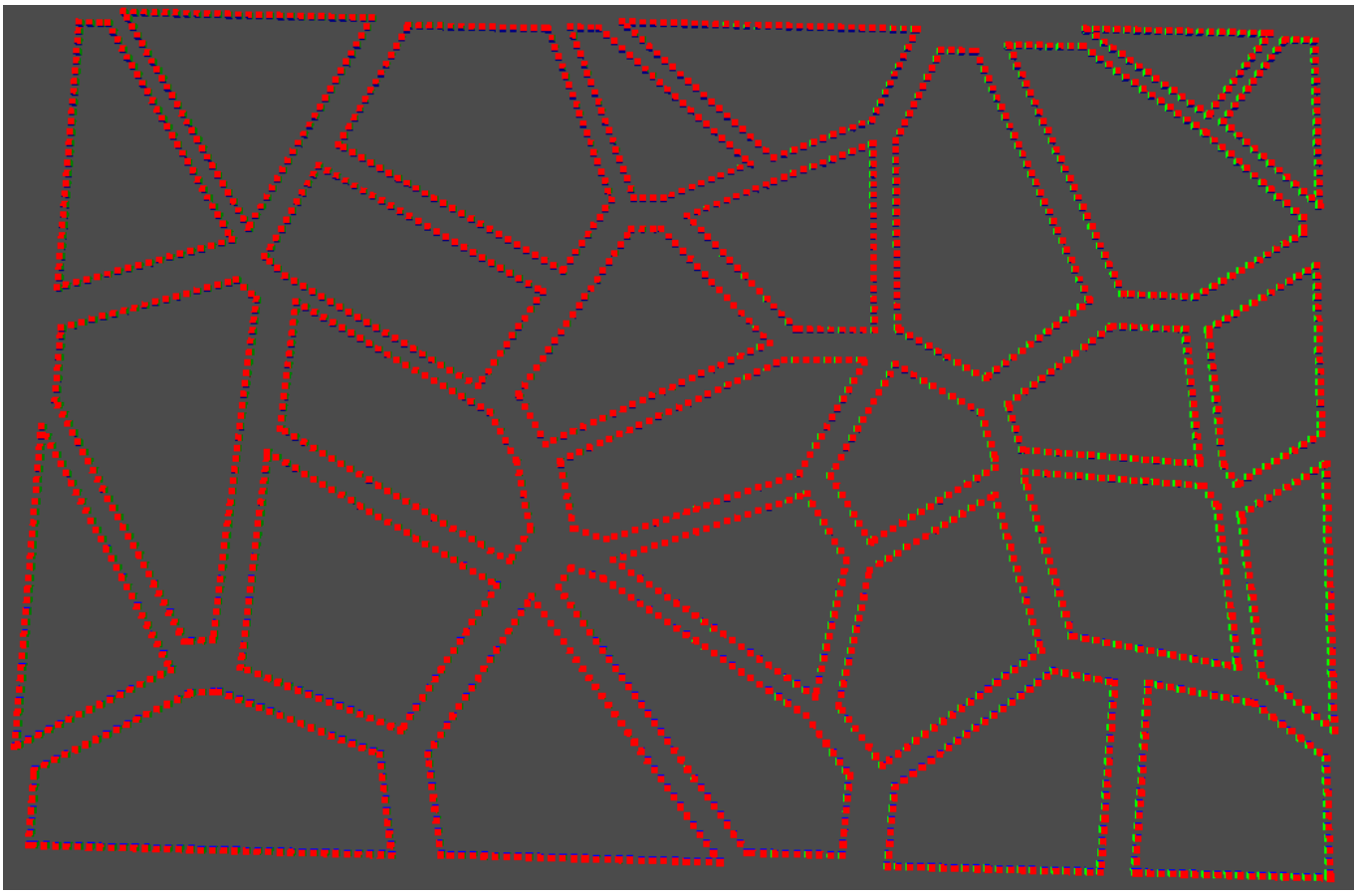
Was able to scale the polygons down, but had issues





Got scaling to work :)

Every voronoi shape has a random edge chosen, this edge is placed at the middle of the polygon and cuts it in half.
Unless it is too small or has < 3 verts.



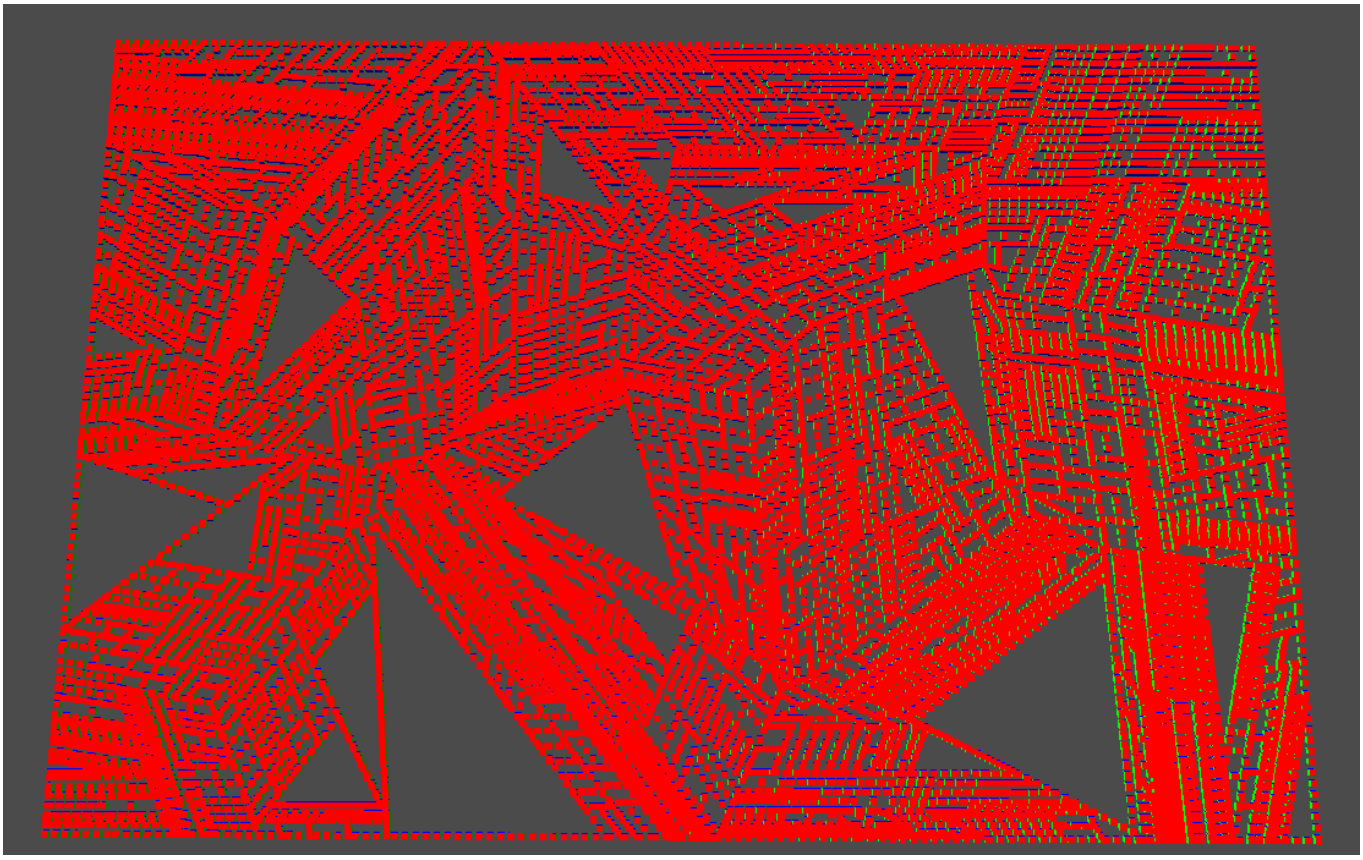
Cut the polygons into grids
... Find longest edge in the polygon and use that as the grid axis
Cut the polygon up into a grid shape.

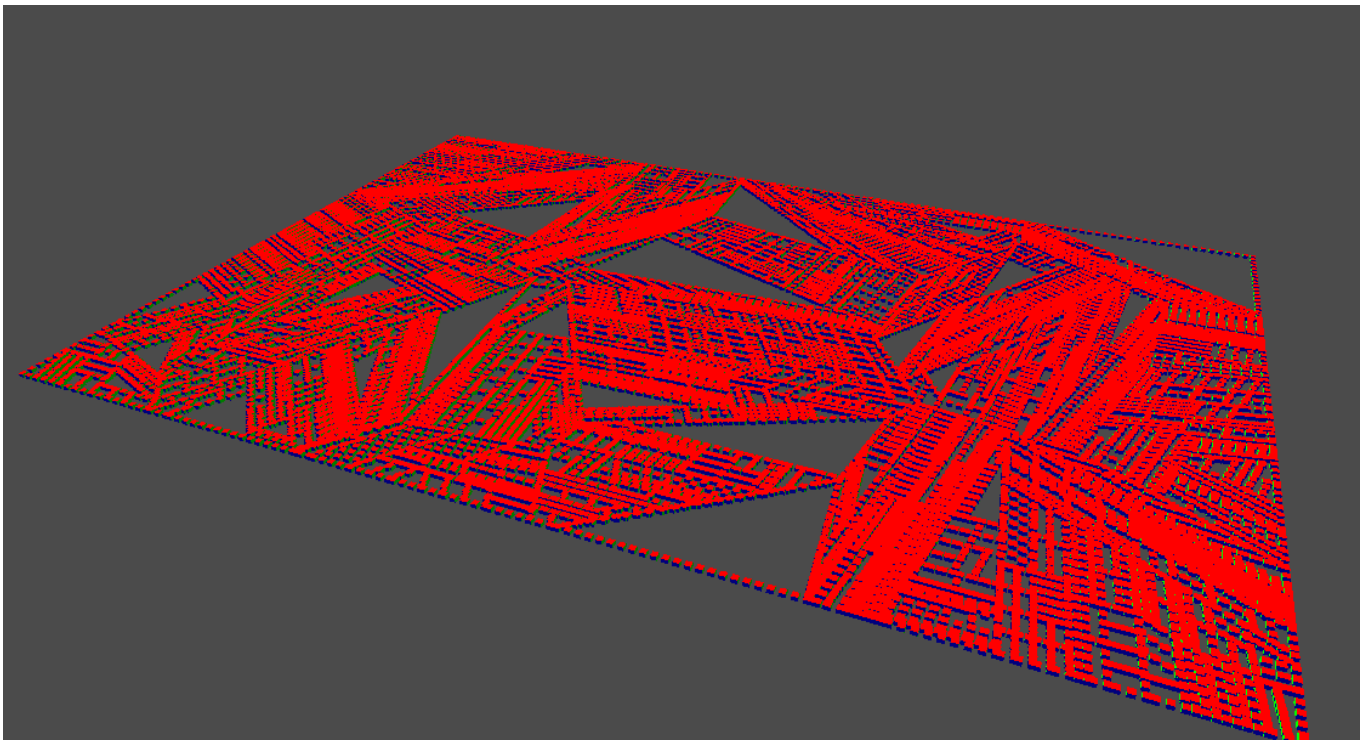
Thought process:

- Find longest edge and return as line: l1
- Find perpendicular of this line: l2
- Move l1 perpendicularly by a random amount and cut the polygon, store the CUT polygon and keep iterating over the other one until it is null.
- Move l2 perpendicularly by a random amount and cut the polygon in the newlist, store the Cut polygon and keep iterating over the other one until it is null.

??- how does line know which direction to travel?

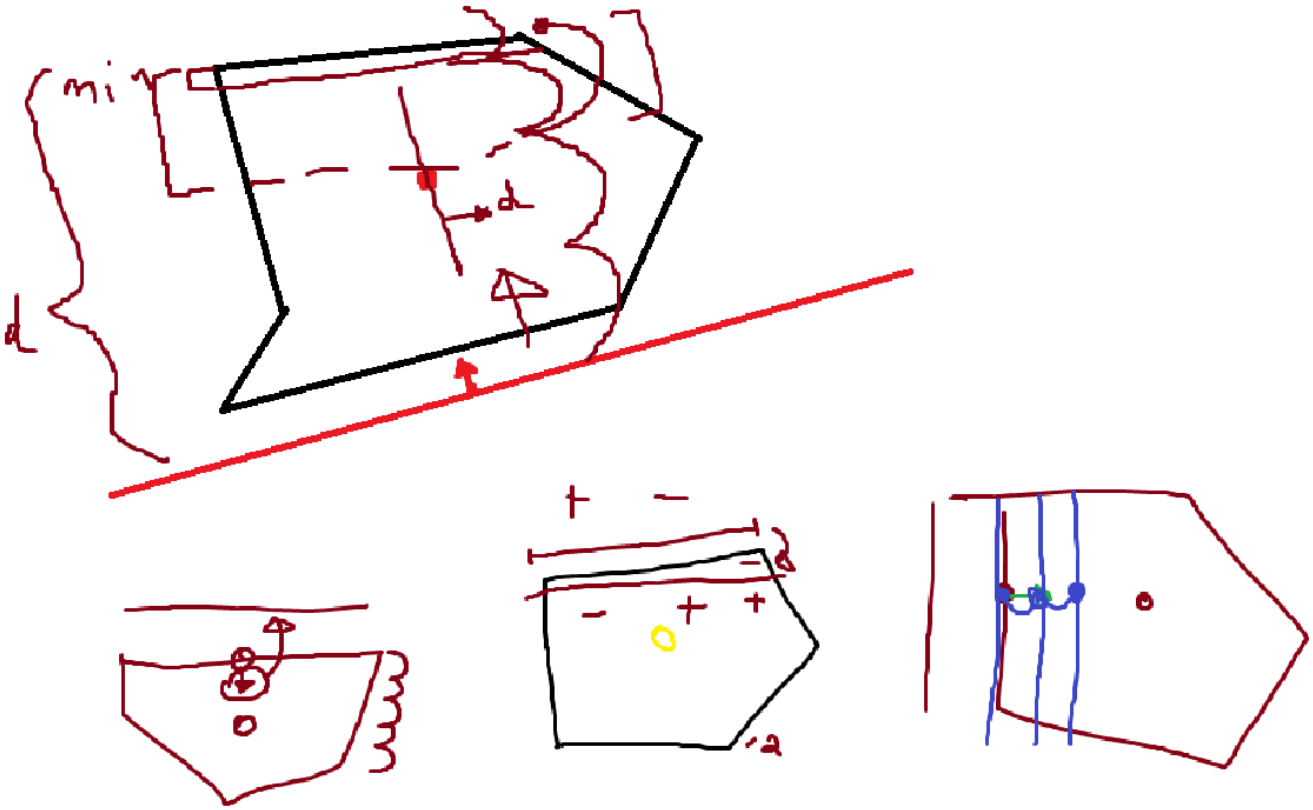
Messed around with some numbers and ended up with something kinda abstract?



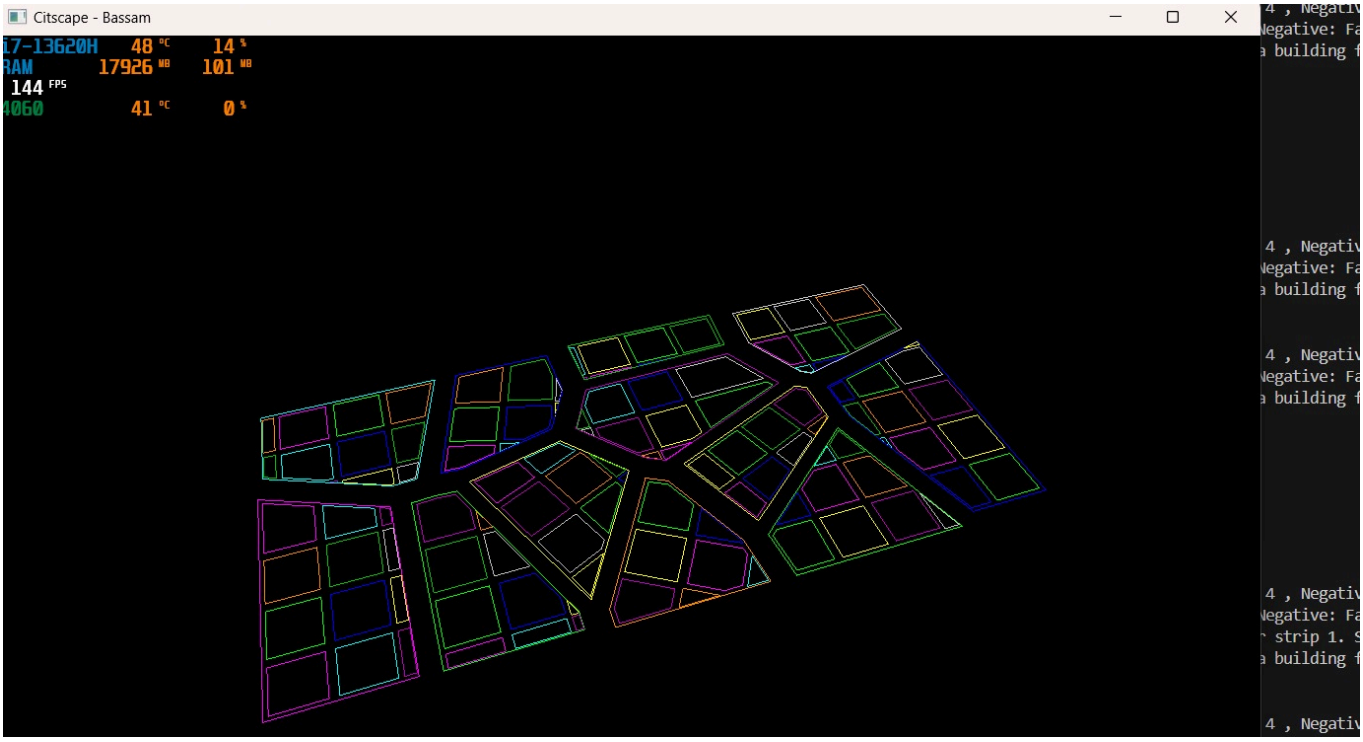


Use GlDrawLines to create roads? -raj

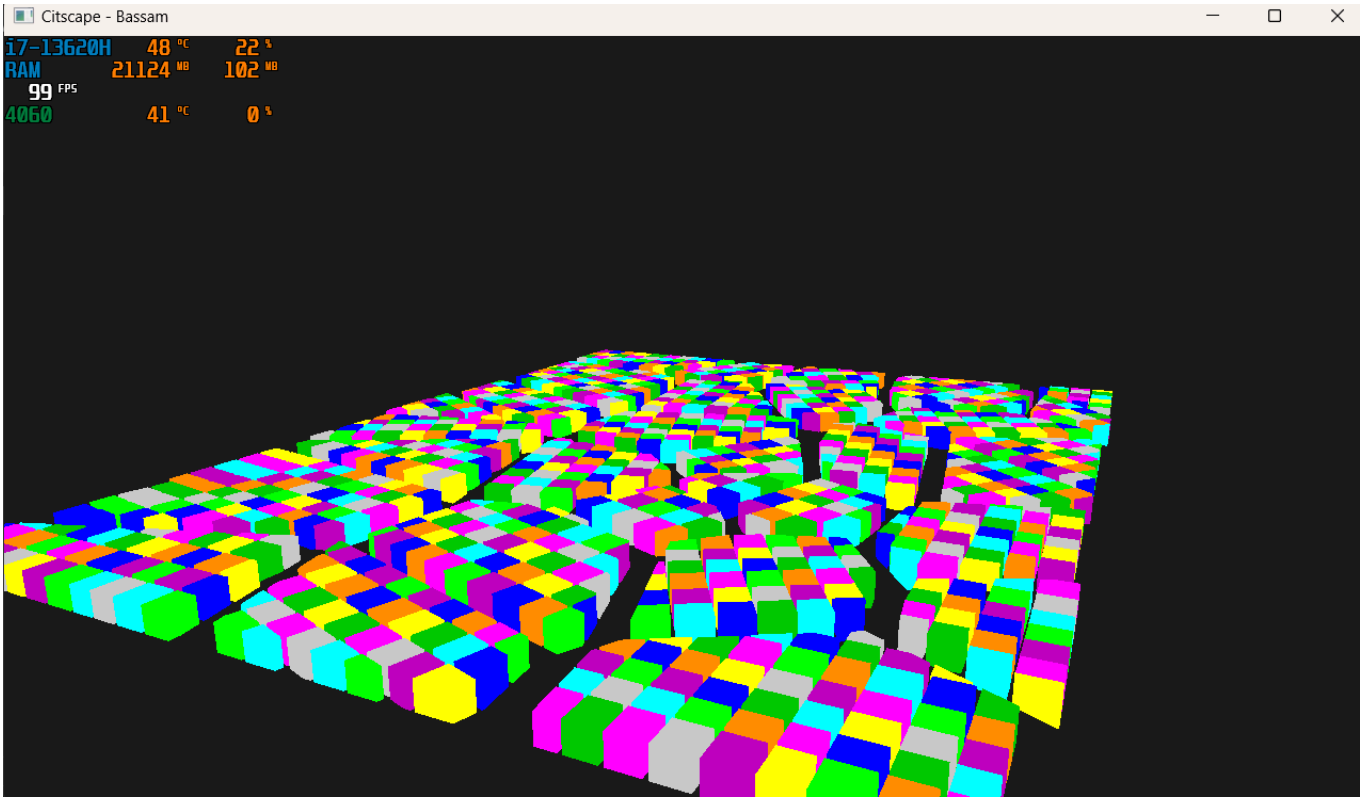
Polygon sweeping logic, Line travels across polygon and clips it using previously created clipping logic. Then go to the center, move perpendicular to starting point by moving around the distance of the longest edge. and sweep through again cutting the streaks of polygons into smaller building sized ones.



Managed to fix and modulate my polygon sweeping logic:



Fixed up Refactored sweeping and clipping code.



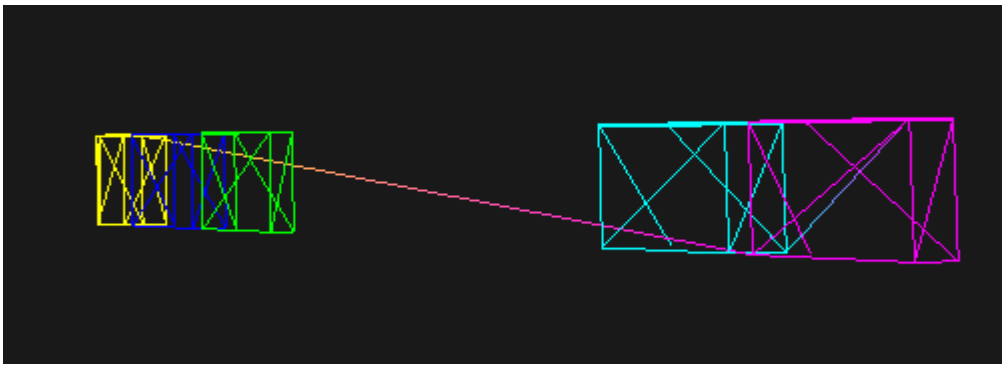
Having roof issues now, Might create some time type of formula for building complex shapes with triangles...

A couple ideas moving forward:

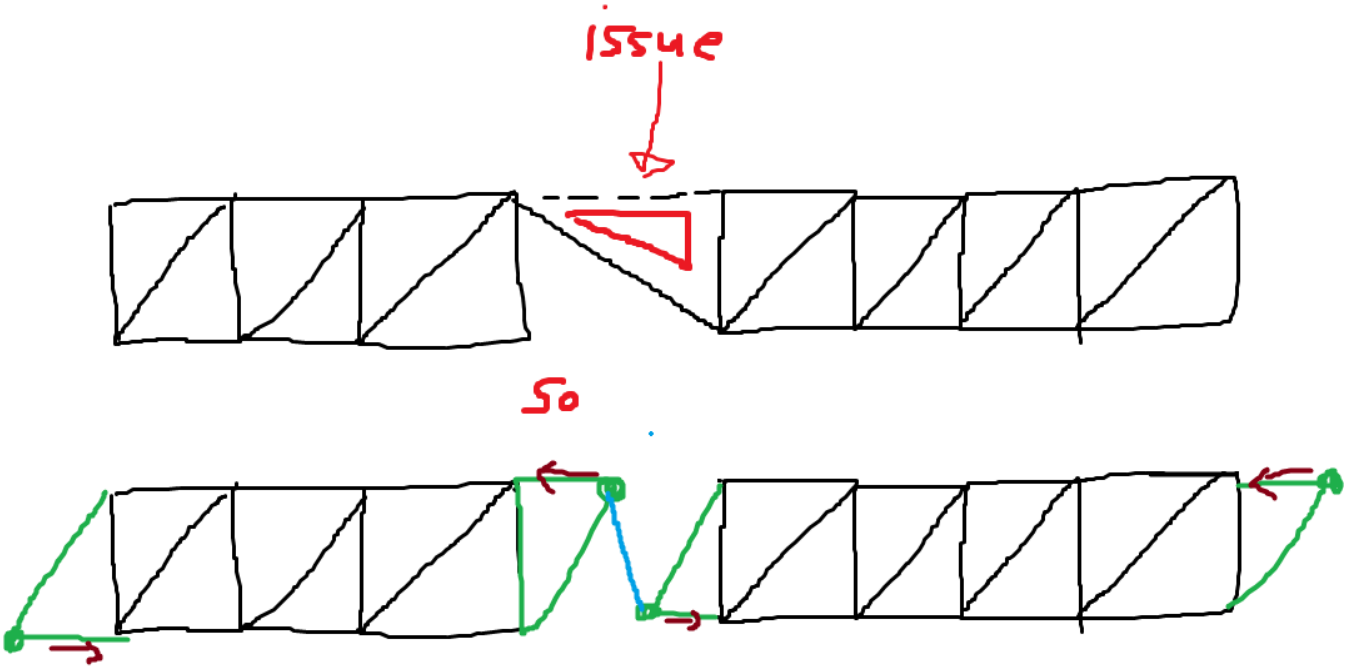
- 3 vert polygons = 1 vert will be displaced lower to resemble

- Same building duplicated in place, each duplicate will be moved up and scaled down (similar to pic attached)

Triangulation / vertex index error...



So the way I triangulate the roof of a building and the walls at the same time , I end up connecting them as shown above,
For me to make the triangle invisible I had to add 1 vertex in the beginning & end... then put the two verticies together (at the end & beginning of a loop); so it becomes an invisible line.



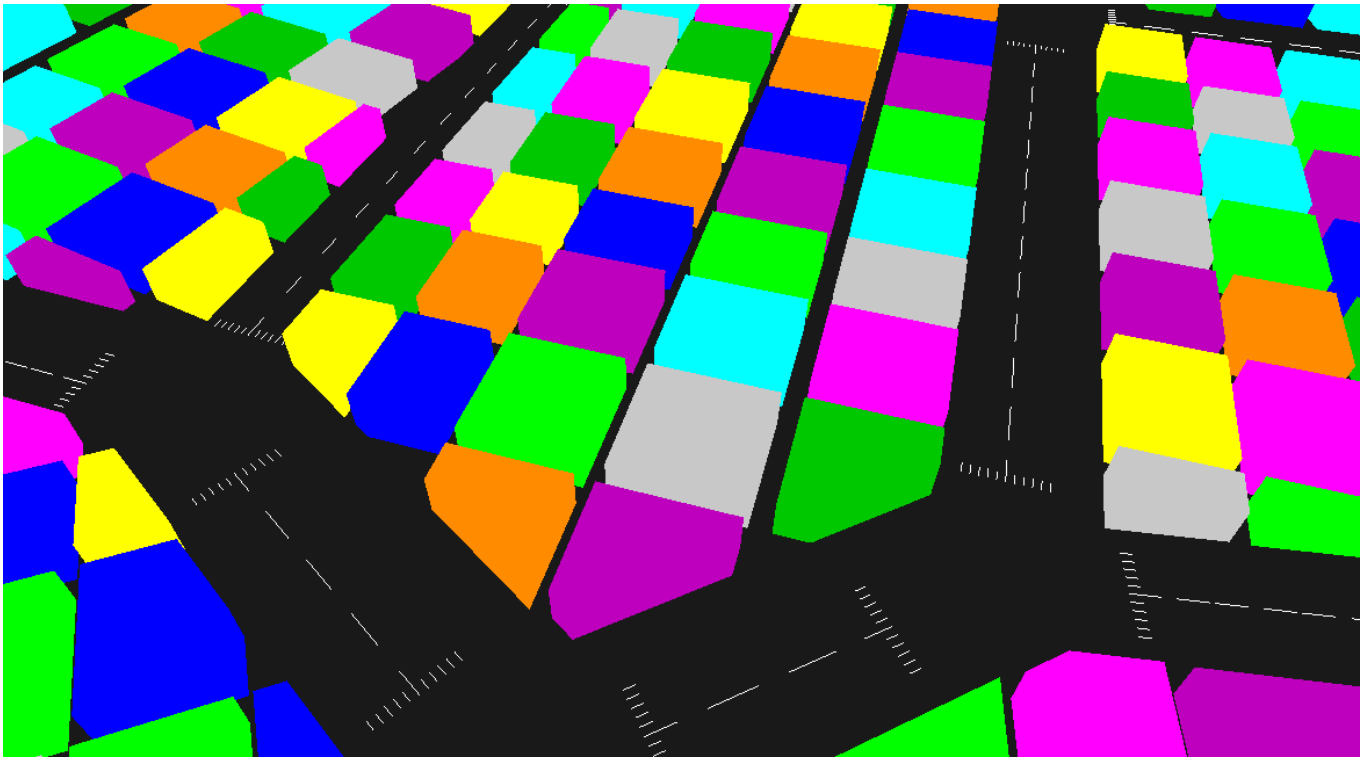
Adding the two verticies in the beginning and moving them into the first actual vertex of the building would leave us with just an invisible line instead of a triangle 😀 😀 😀

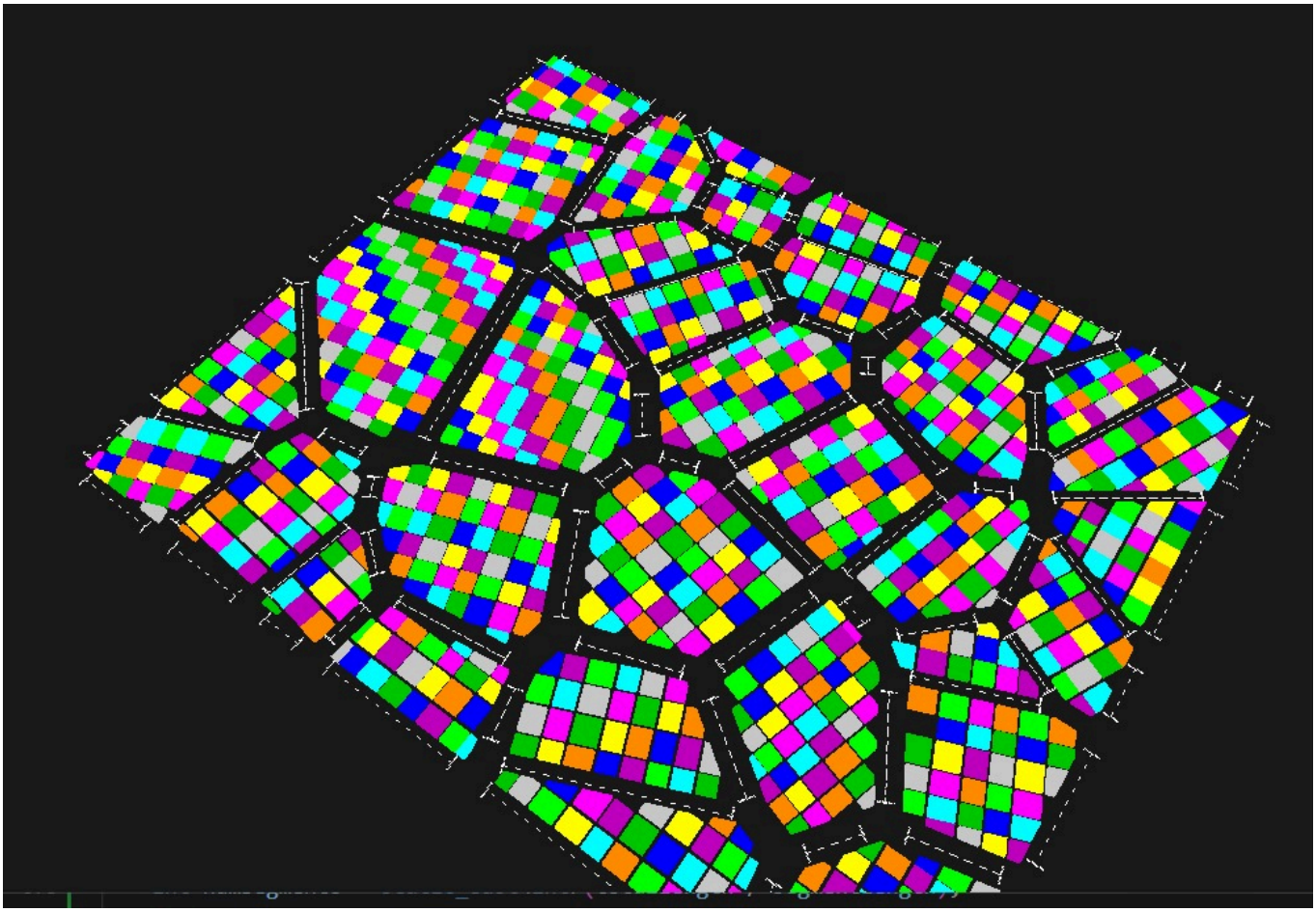
Roads = m_voronoiCells
side walk = m_chunks scaled

Road thoughts:

- Every edge is usually connected to something...
- Assume x distance from start and end of edge is an intersection.
  - p1 to p2 = distance
  - Remove (amount) from each side
  - d

Crosswalk:





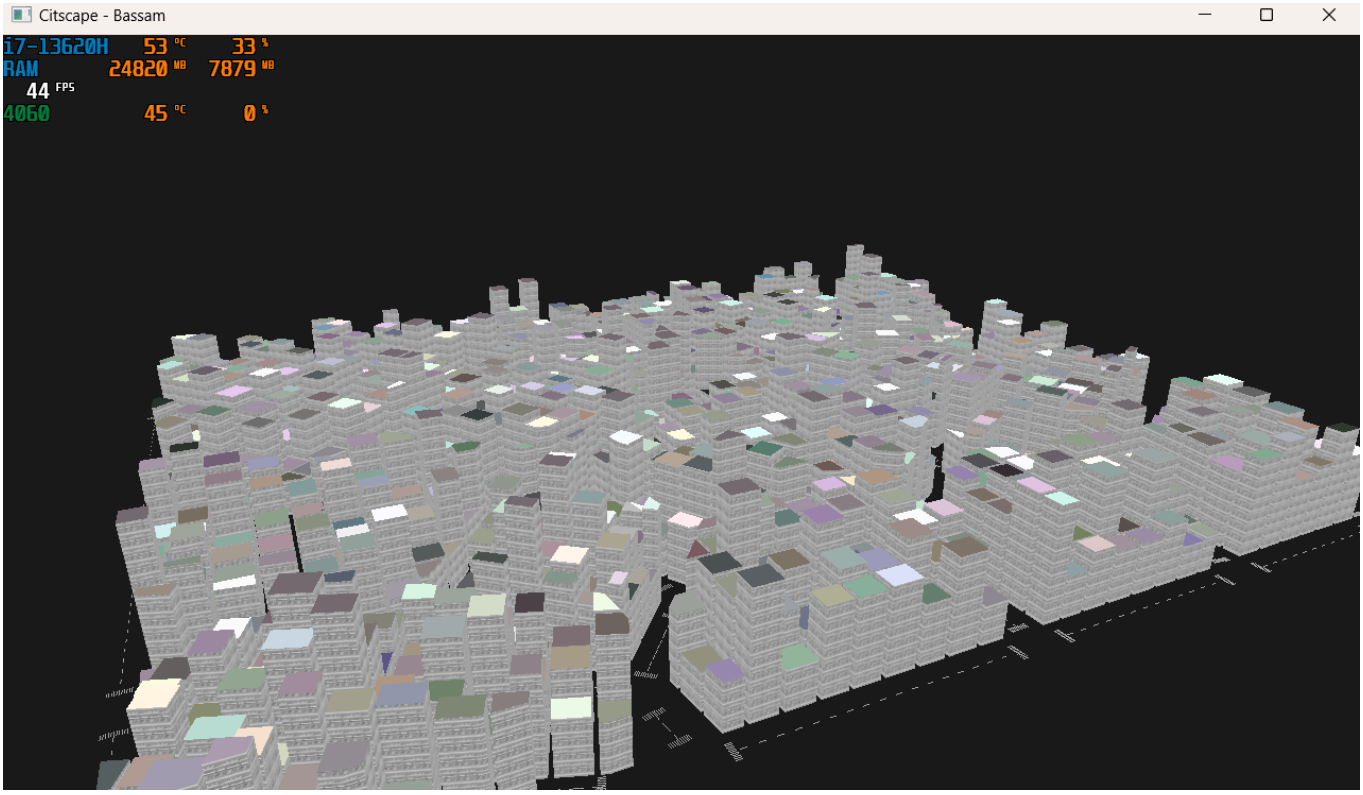
Trouble texturing the buildings:

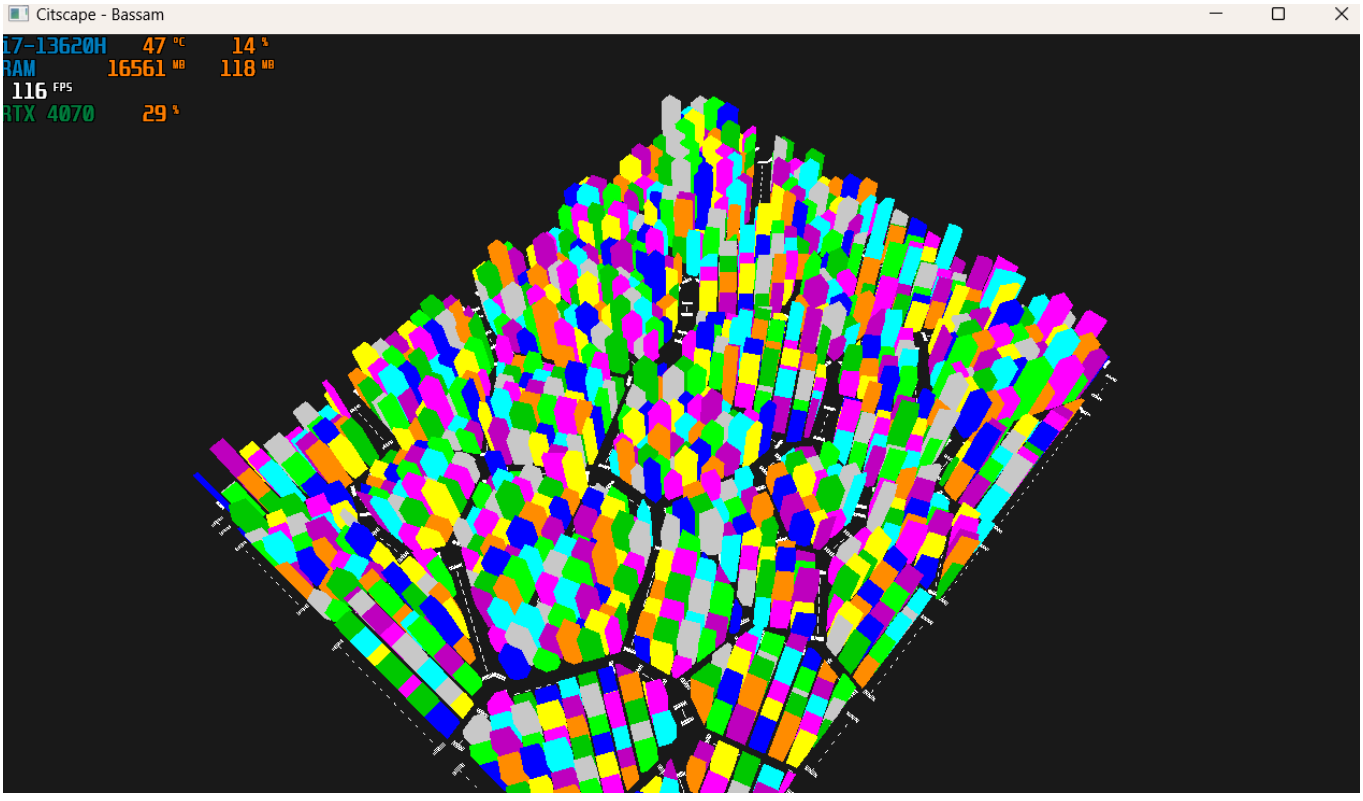
01 11

00 10

Should tile based on triangle height

HUGE ISSUE MOVING UV DATA TO SHADER
Always Use offsetof for Attribute Offsets from now on:
Hardcoding offsets (3 * sizeof(GLfloat)) is error-prone because it assumes or has aligment issues.





Had a huge issue getting .tga to crop
downloaded paint.net to convert .tga cuz all other converters make my shit tilt

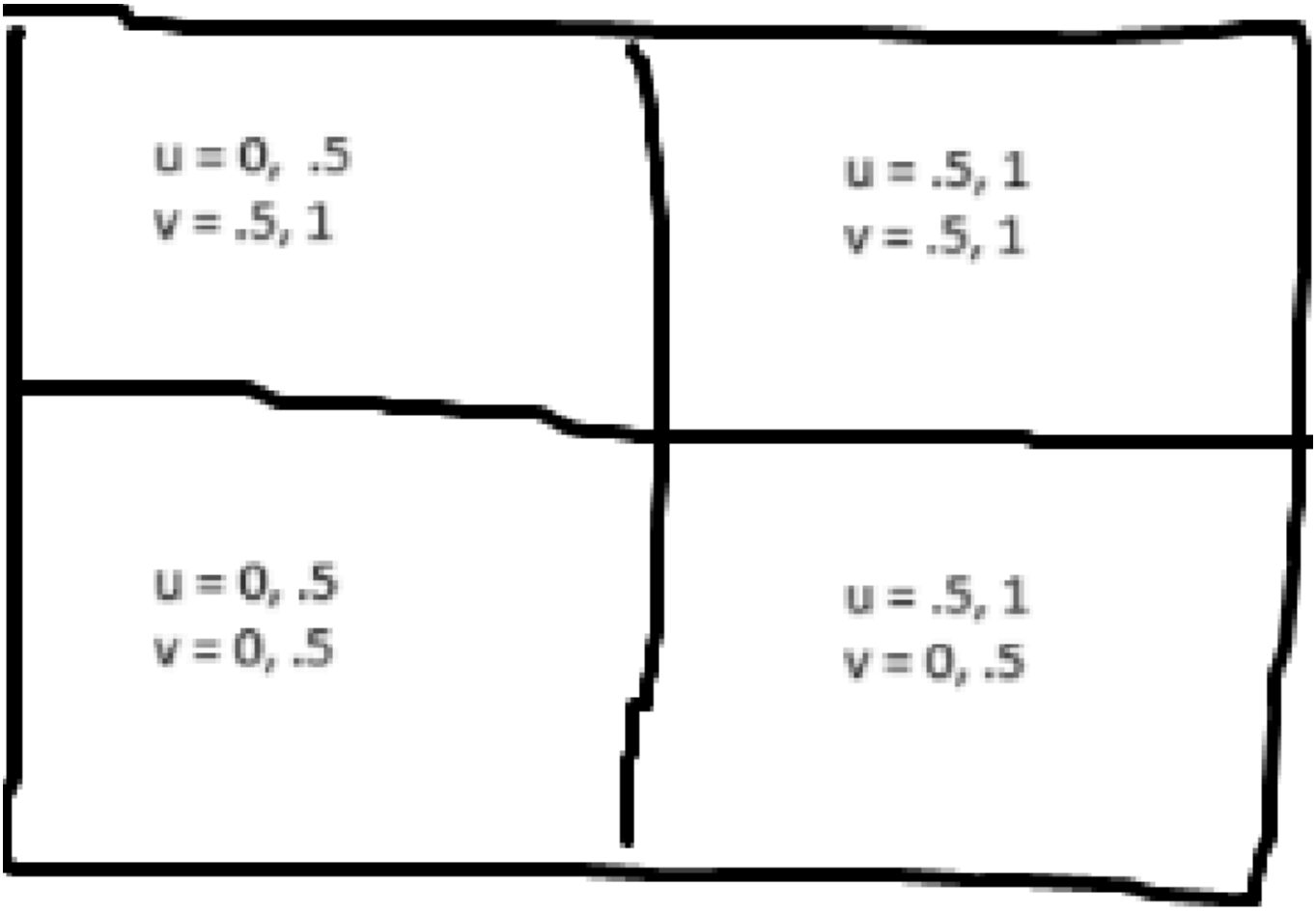
```
```GL, by default, expects rows of pixels to be padded to a multiple of 4 bytes. A 1366 wide texture with 1 byte or 3 byte wide pixels, will not be naturally 4 byte aligned.

Possible fixes for this are:

- Tell GL how your texture is packed, using (for example) `glPixelStorei(GL_UNPACK_ALIGNMENT, 1);`
- Change the dimensions of your texture, such that the padding matches without changing anything
- Change the loading of your texture, such that the padding is consistent with what GL expects
```

After hours of debugging all I had to do was convert to 32bit (takes more space but has the needed byte padding to texture normally :D)

Next ima try texture atlasing different buildings and tilting


```

```
float tileUmin = 0.5f;

float tileUmax = 0.0f;

float tileVmin = 0.5f;

float tileVmax = 0.0f;



//Top left

// float tileUmin = 0.0f;

// float tileUmax = 0.5f;

// float tileVmin = 0.5f;

// float tileVmax = 1.0f;



//Top right

// float tileUmin = 0.5f;

// float tileUmax = 1.0f;

// float tileVmin = 0.5f;

// float tileVmax = 1.0f;

//Bottom Left

// float tileUmin = 0.0f;

// float tileUmax = 0.5f;
```

```
    // float tileVmin = 0.0f;

    // float tileVmax = 0.5f;



    //Bottom Right

    // float tileUmin = 0.5f;

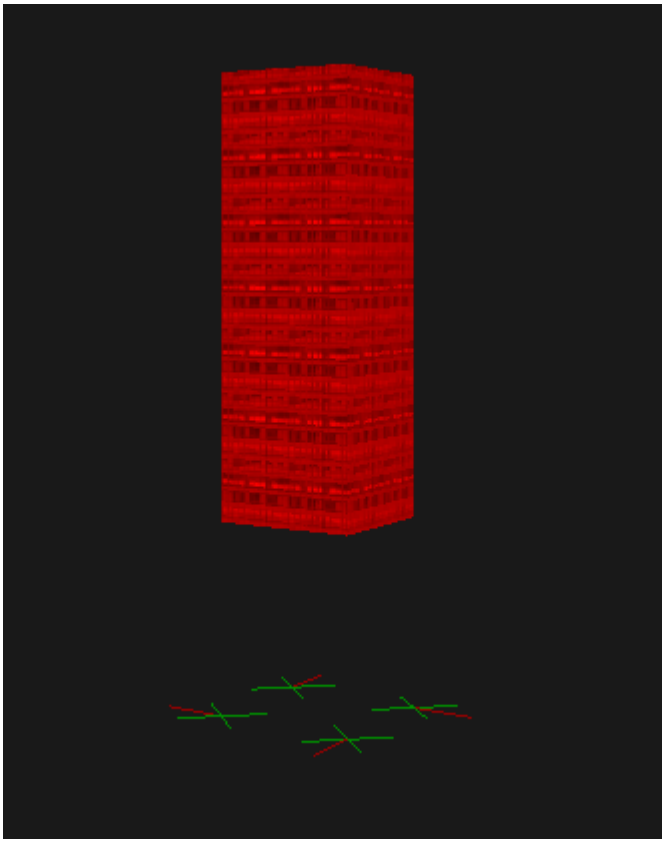
    // float tileUmax = 1.0f;

    // float tileVmin = 0.0f;

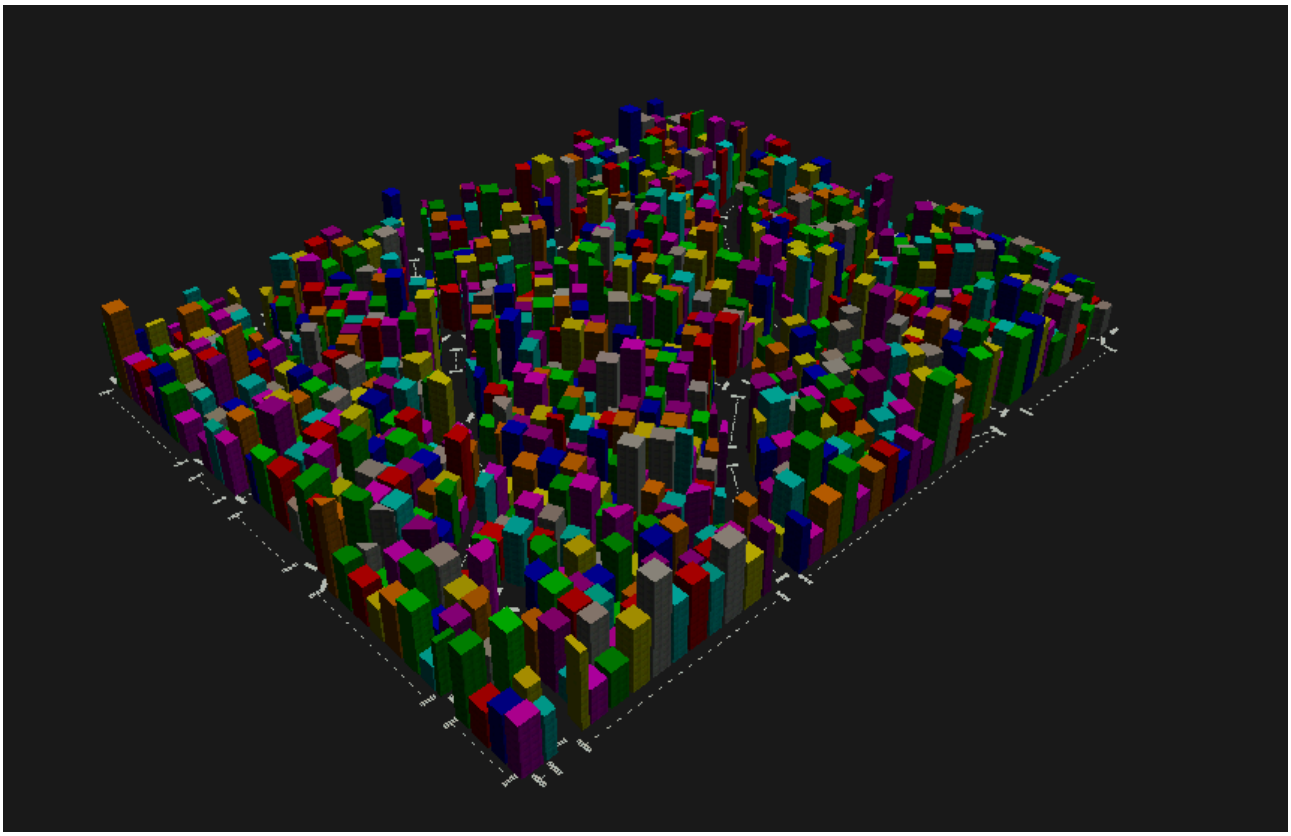
    // float tileVmax = 0.5f;
```

Biggest issue yet...
Adding normals to the verticies breaks everything, simply sending the normal to the fragment shader messes it all up... I've been trying to figure out how for the longest time.

Manually calculated normals:



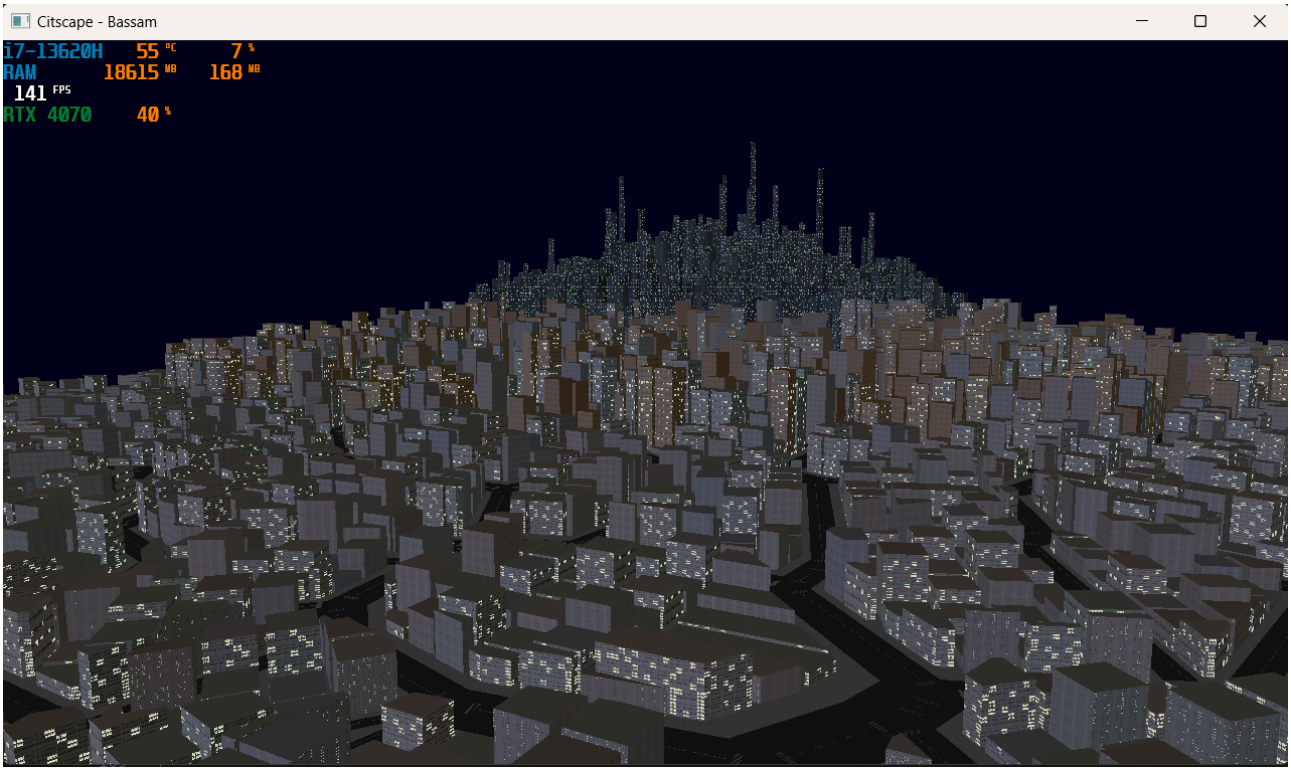
- Add lighting - DONE

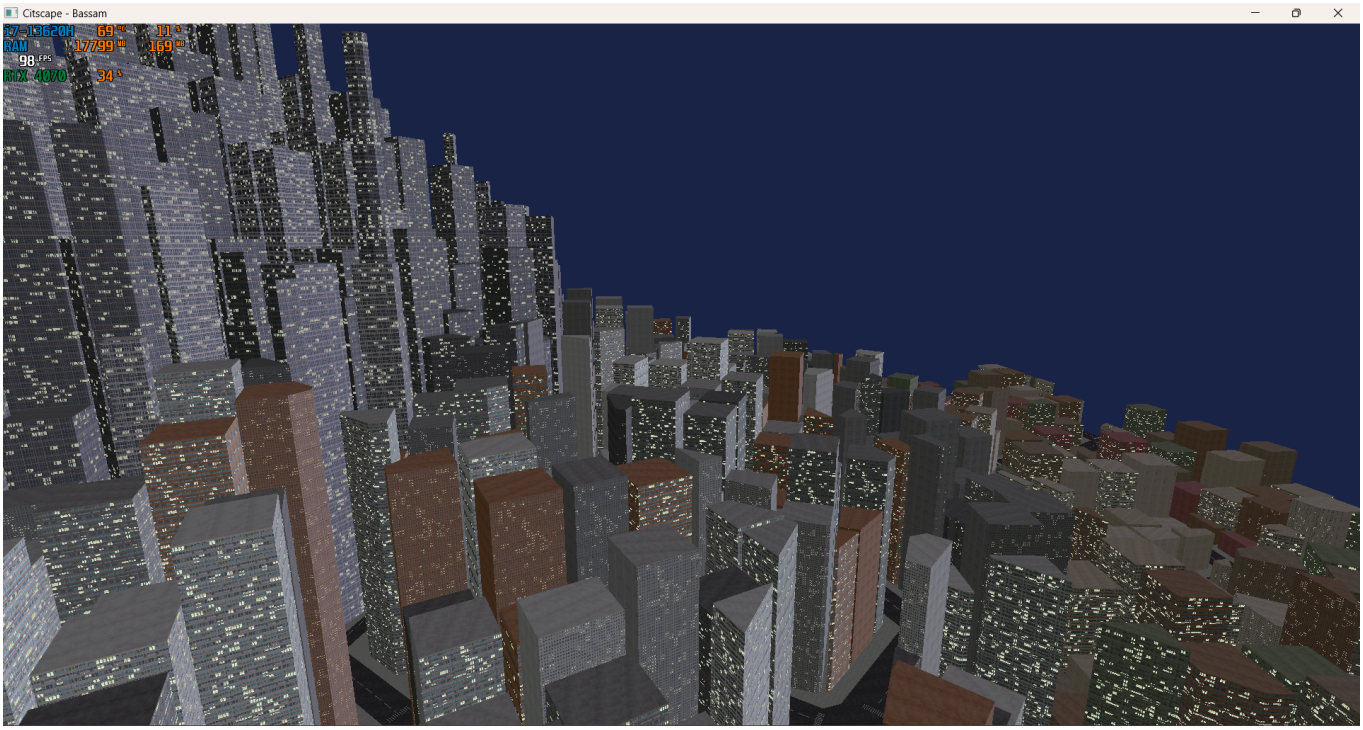


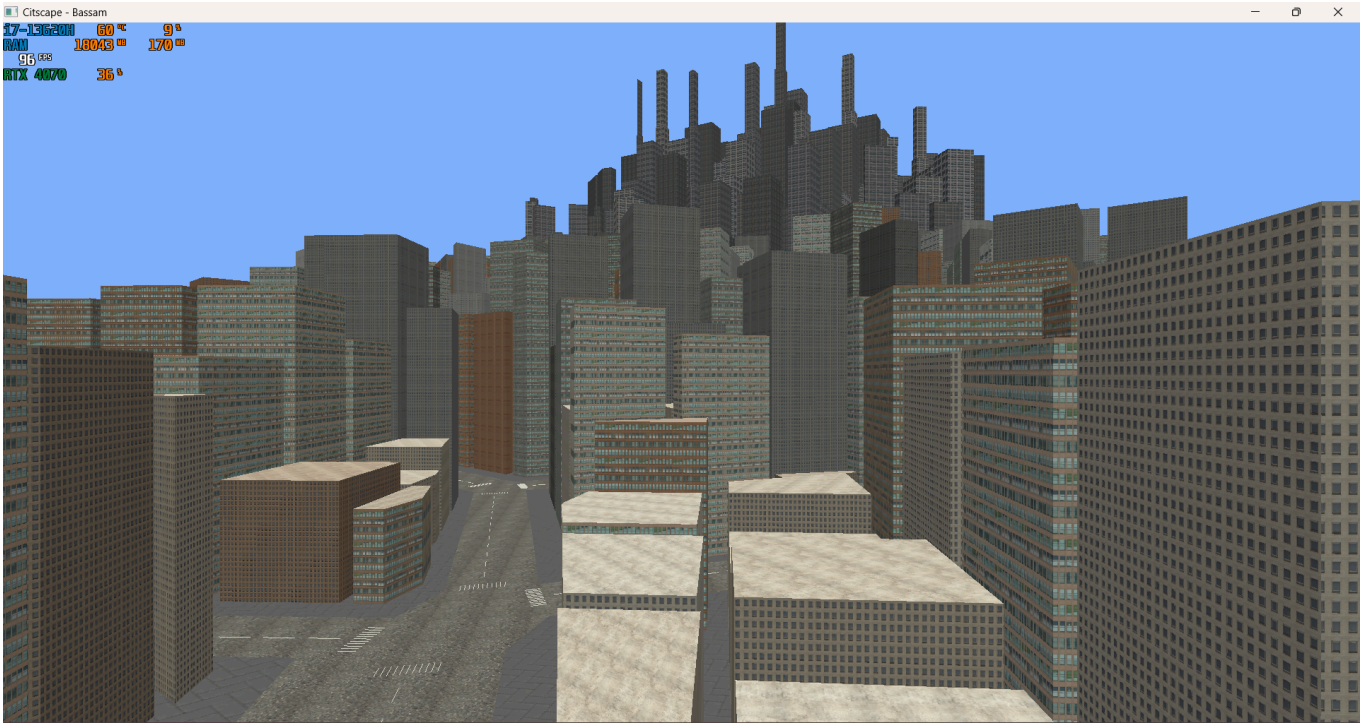
DONE:

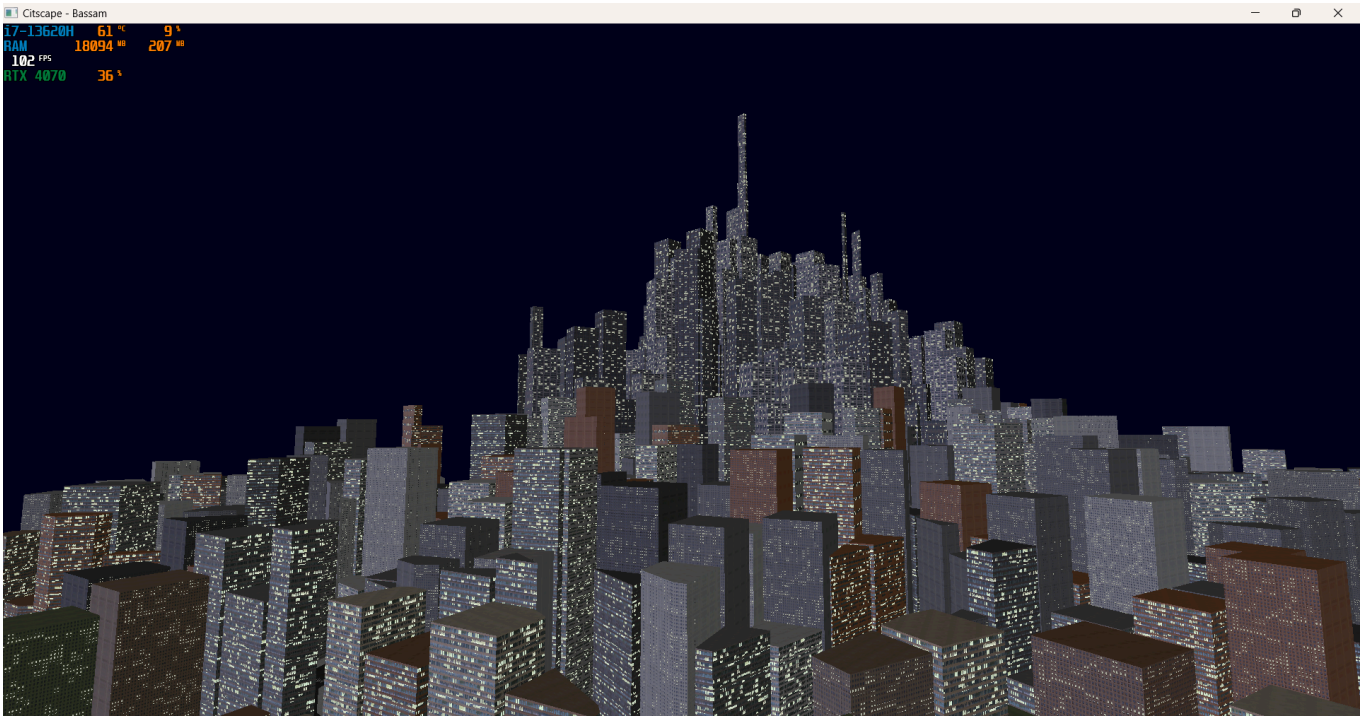
- Change texturing

    - Give buildings random colours - DONE
    - Overlap tiling windows ontop - DONE
- Different textures for buildings - DONE
- Sidewalk - DONE
- Refactor code to use wolf **DONE**
- Day & Night cycle - DONE
- use the x and z plane to generate a UV map - DONE
- Different textures for buildings - DONE
- Have clearColor increment to create blue background - DONE
- Fix building window lights - DONE
- Add building types according to districts
    - Building struct [polygons, textureLayer, district] - DONE
- Refactor code for readability - DONE
- Thin buildings shoud not render - DONE
- Fan triangulation textures (use scalability?) - DONE
- Change building heights based on areas - DONE
- Add special building (tower) - DONE
- add an option to InvertY on the mouse control, perhaps a simple button toggle for your camera - DONE




Citscape - Bassam
i7-13620H 69 °C 11 %
RAM 17799 MB 169 MB
98 FPS
RTX 4070 34 %


Citscape - Bassam
i7-13620H 60 °C 9 %
RAM 18043 MB 170 MB
96 FPS
RTX 4070 36 %


Citscape - Bassam
i7-13620H 61 °C 9 %
RAM 18094 MB 207 MB
102 FPS
RTX 4070 36 %