LIDAR MAP METHOD

This method is very efficient and only requires 2-3 scans for most maps. Some maps like in the following example may require 5-6 scans.

Initial Scan

If we keep the initial scan at the centre, we get the most information possible out of our first scan

However, the centre may be blocked and we might need to start somewhere else

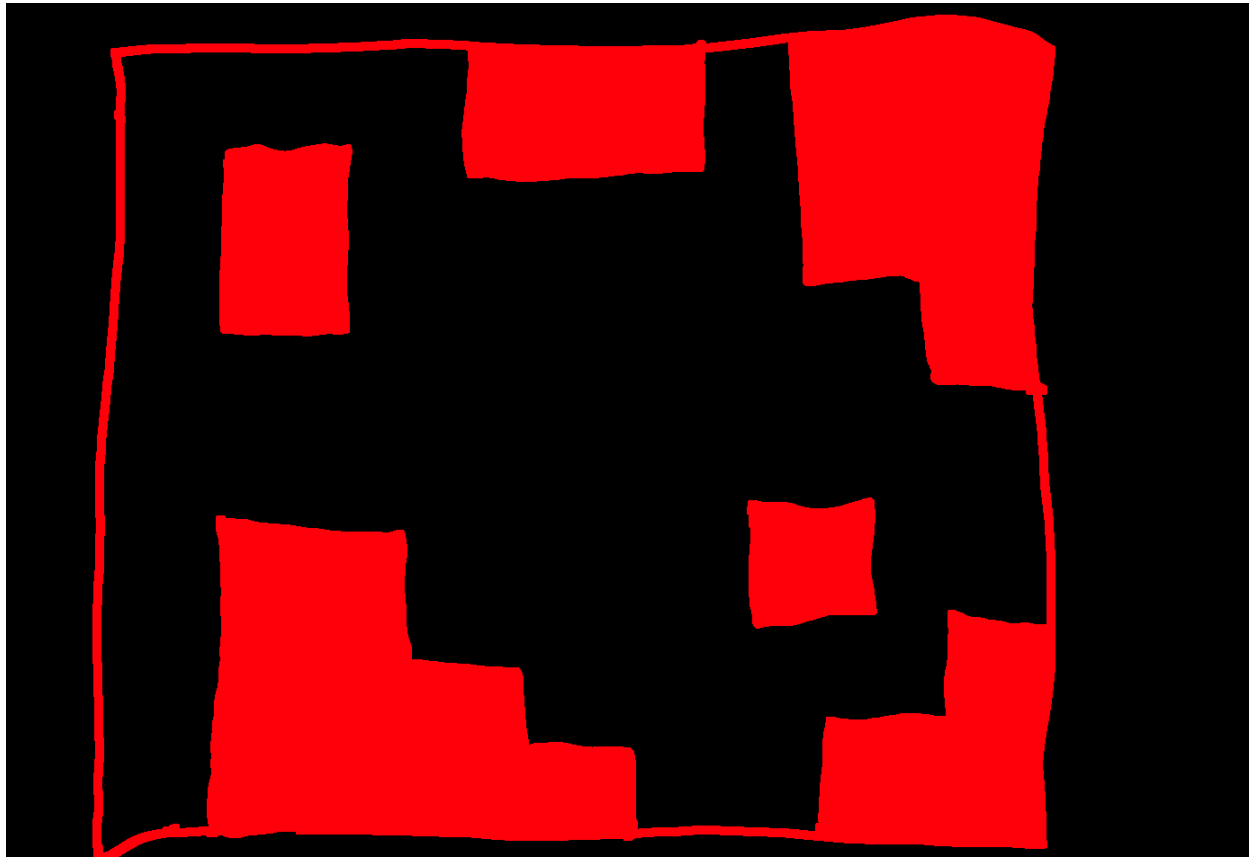
In some maps, the centre may not have the maximum information output

Win condition

- The important point is to find out when to stop scanning and know when the map is done.
- After a bit of thinking, I realised that this happens when all the outside edges of a map are completely filled(forming a square or rectangle depending upon the map shape)
- These edges may be imaginary or real depending on the requirements

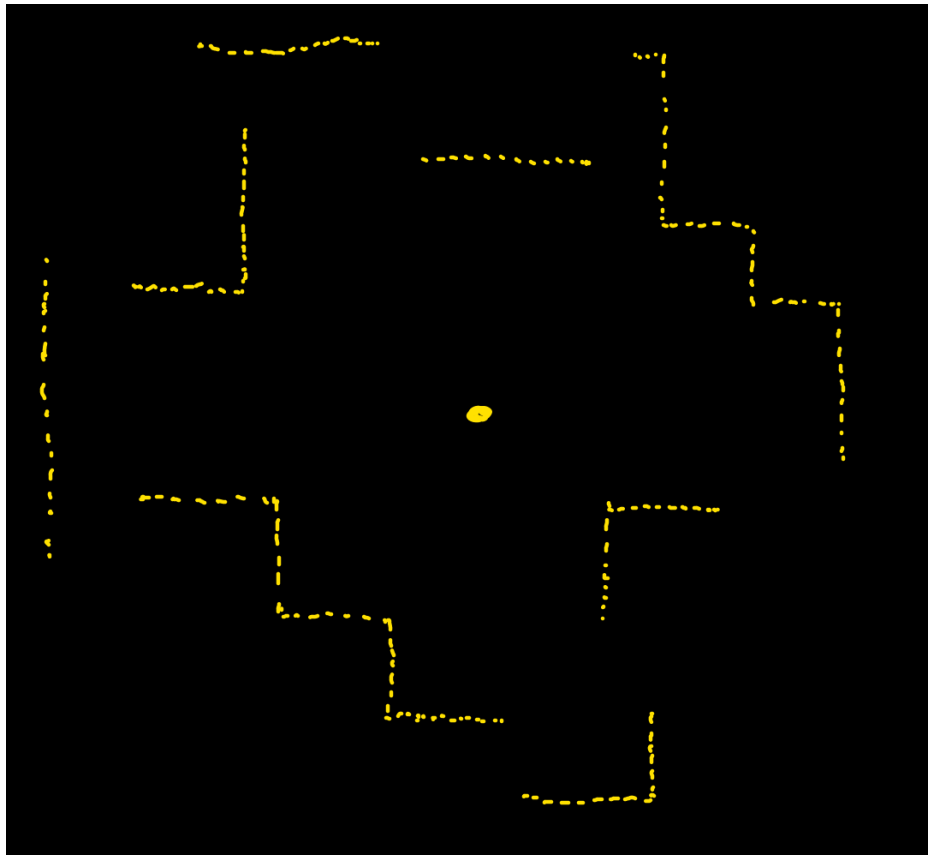
An Example

- We just assume a difficult random map



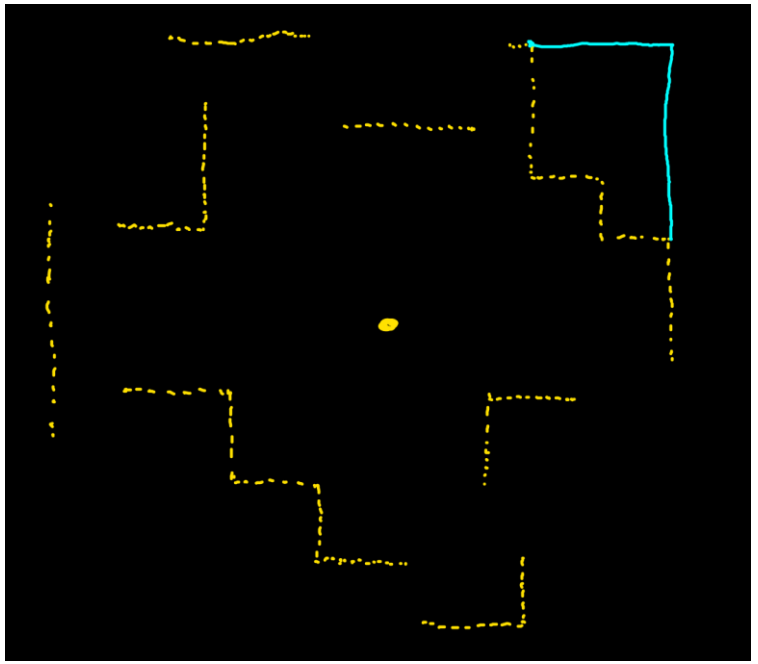
Initial scan at centre

- After performing the initial scan, we get this output



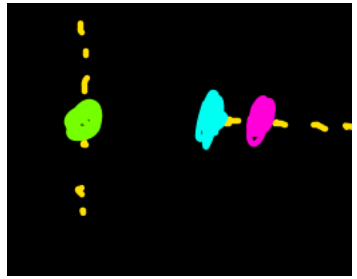
Completing the boundary

- Whenever 2 vertical lines touch the imaginary boundary(represented in light blue), and the 2 vertical lines touch each other indirectly(with a single line or multiple lines), we may complete the boundary between the 2 lines

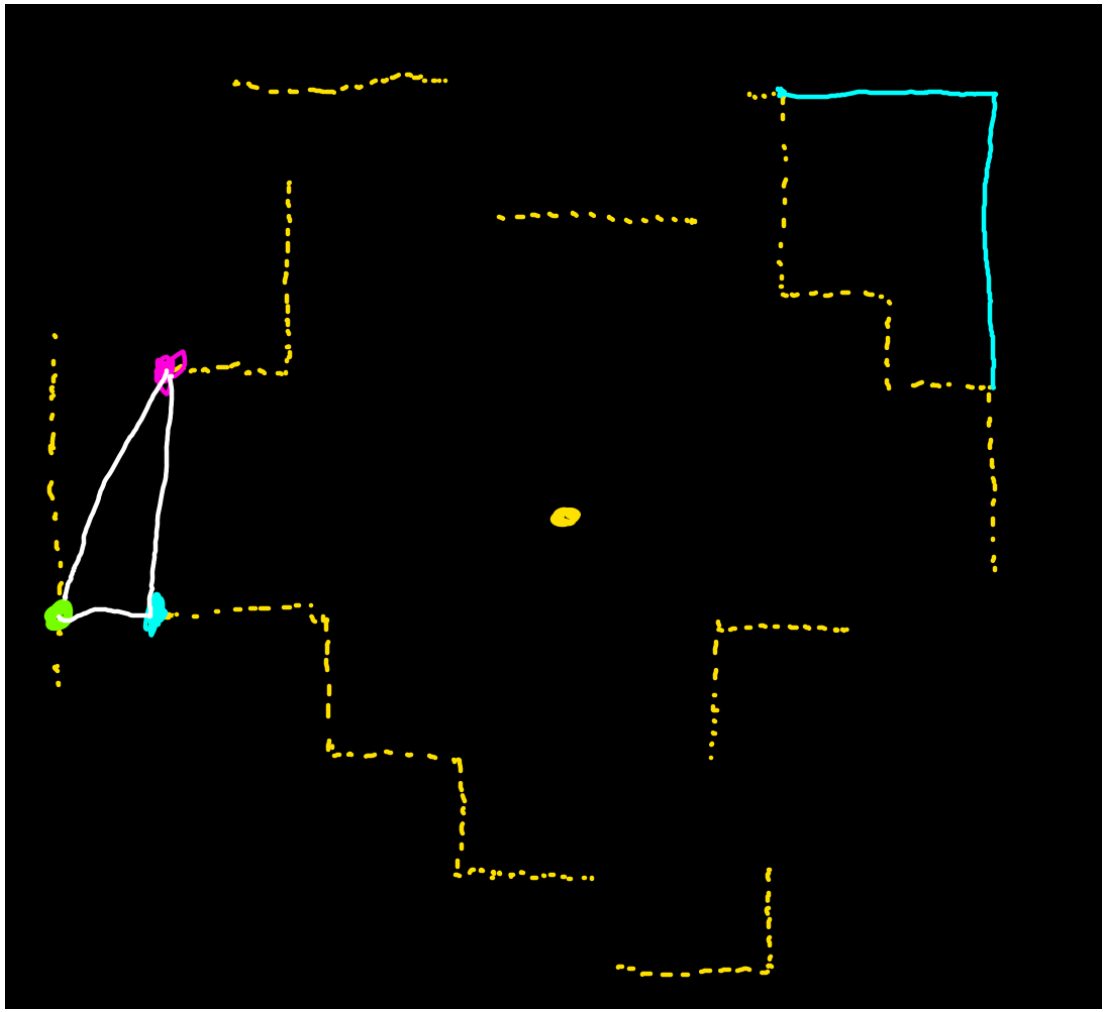


Next Scan

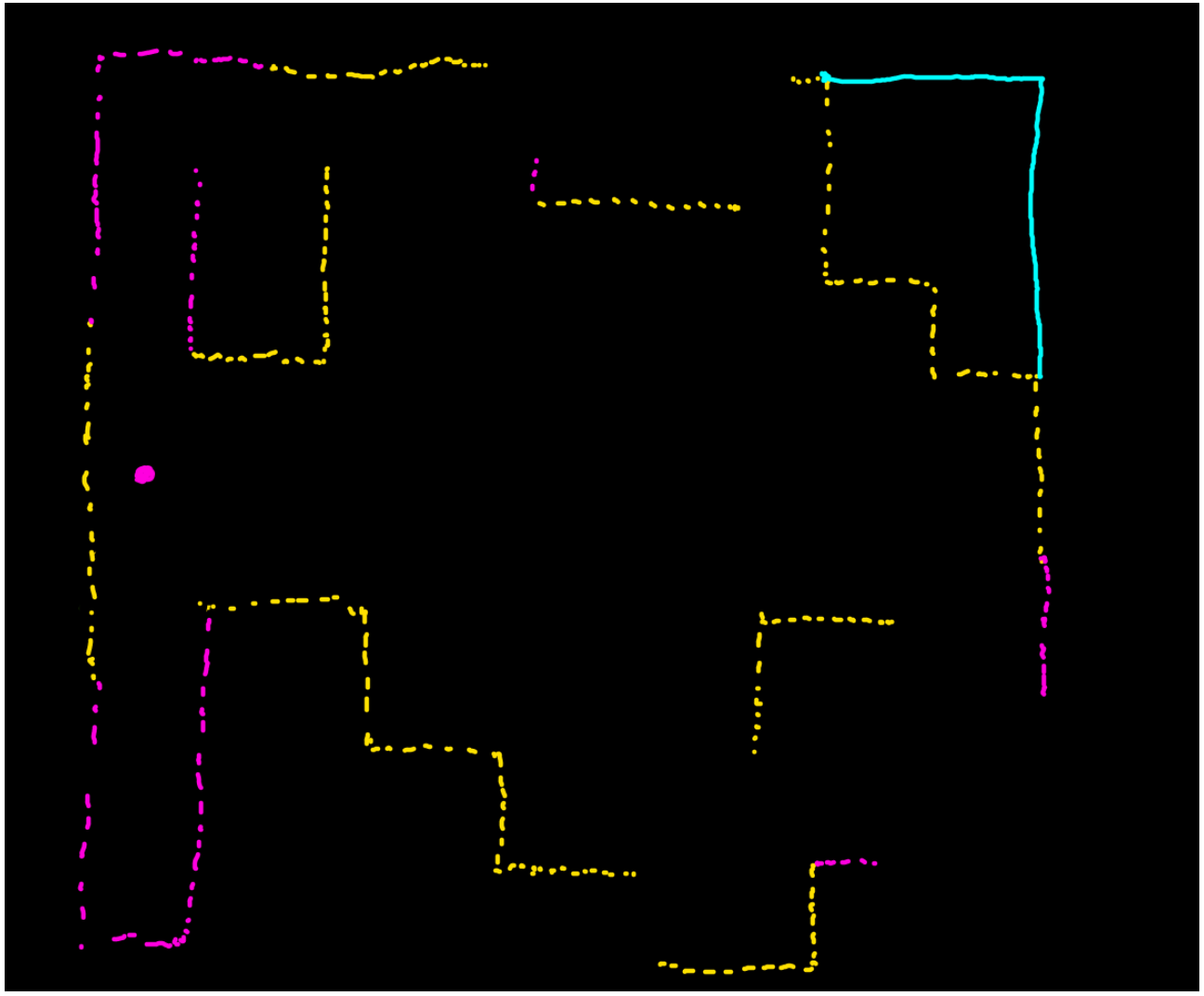
- To find the location of the next scan, we simply find a point that has only 1 point near it and one that is closest to the imaginary boundary. ‘Near’ basically means that the point is less than (r/360) units away from our point since this is the least count of our lidar.
- In this image, light blue is our main point, pink is a point near it and green is a point not near it and is also a point that lies on the imaginary boundary



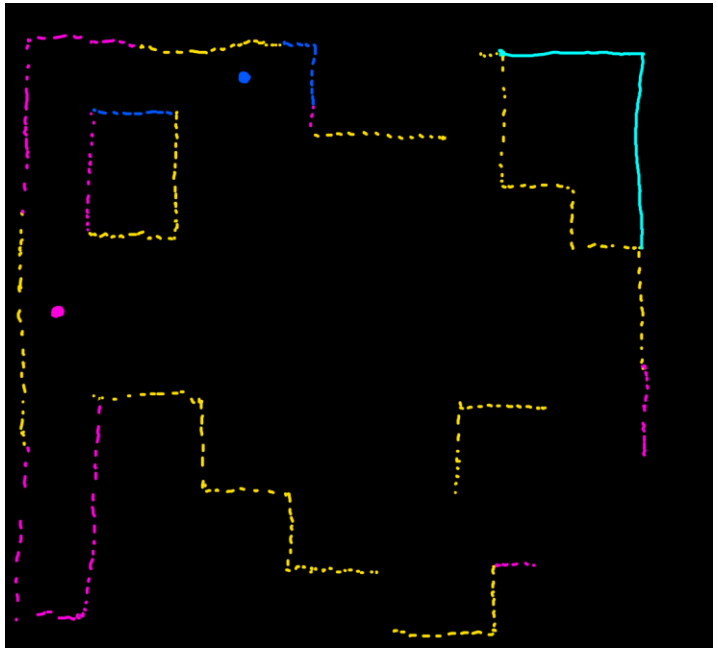
- From the light blue point, we find 2 other points, one on the boundary and one perpendicular to the line joining these 2 points and form a right angle between the 3

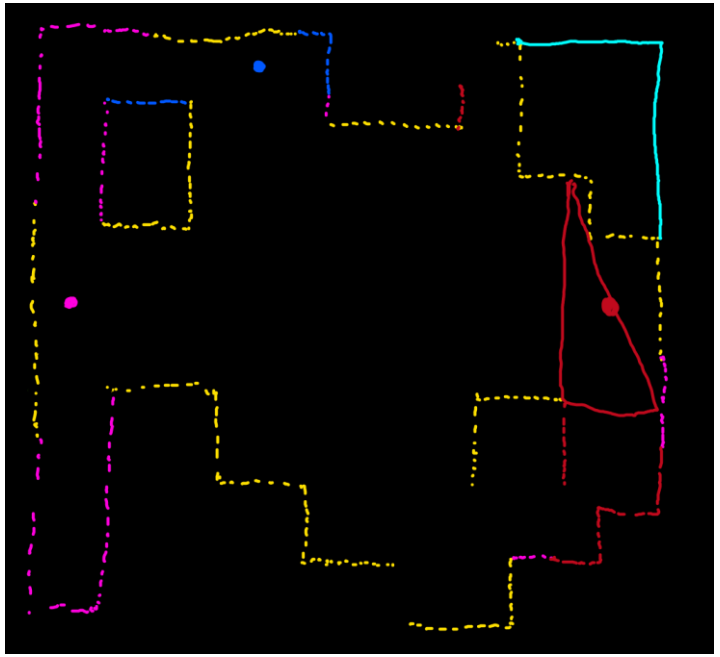


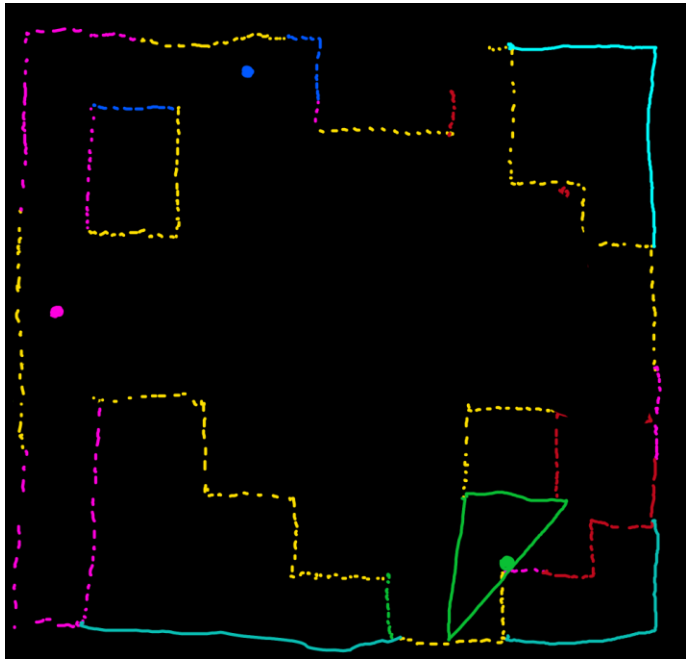
- On the midpoint of the hypotenuse of this right triangle, we run our next scan.(pink)



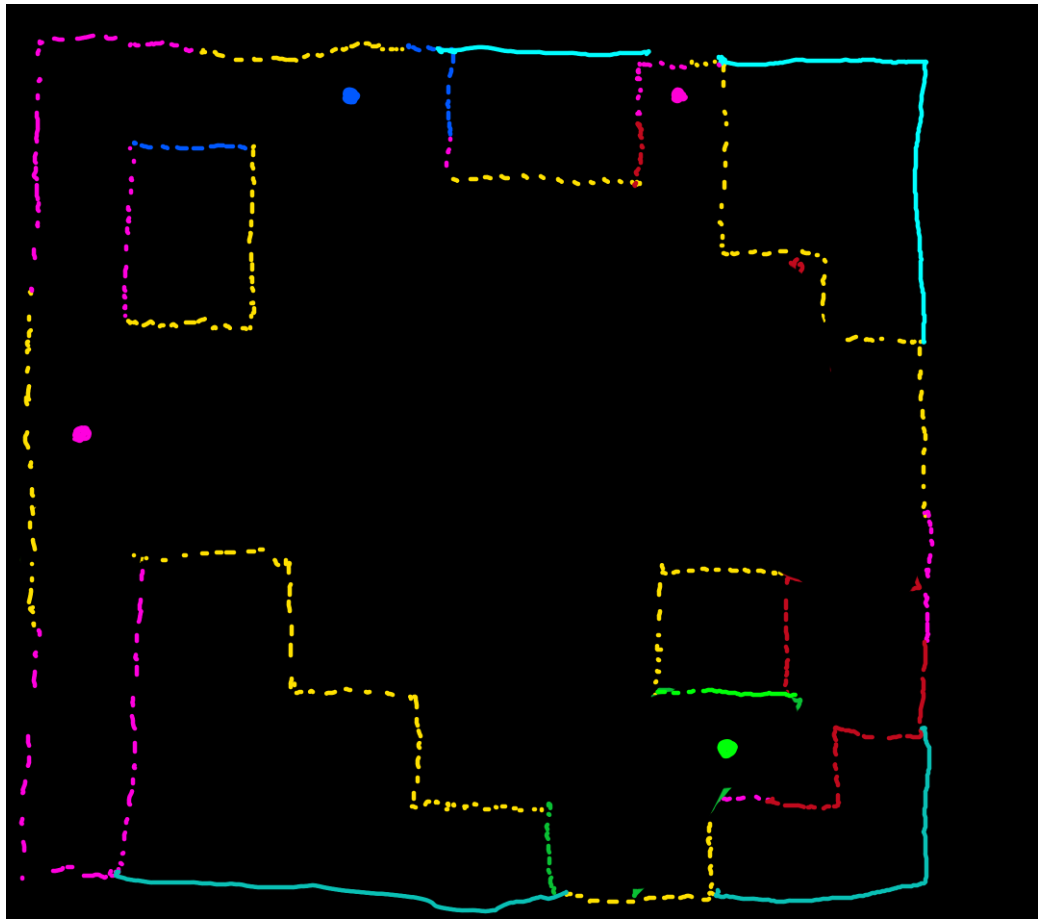
- We keep on adding new scans and try to complete the boundary using the vertical line rule and data from any scans(represented by different colours)



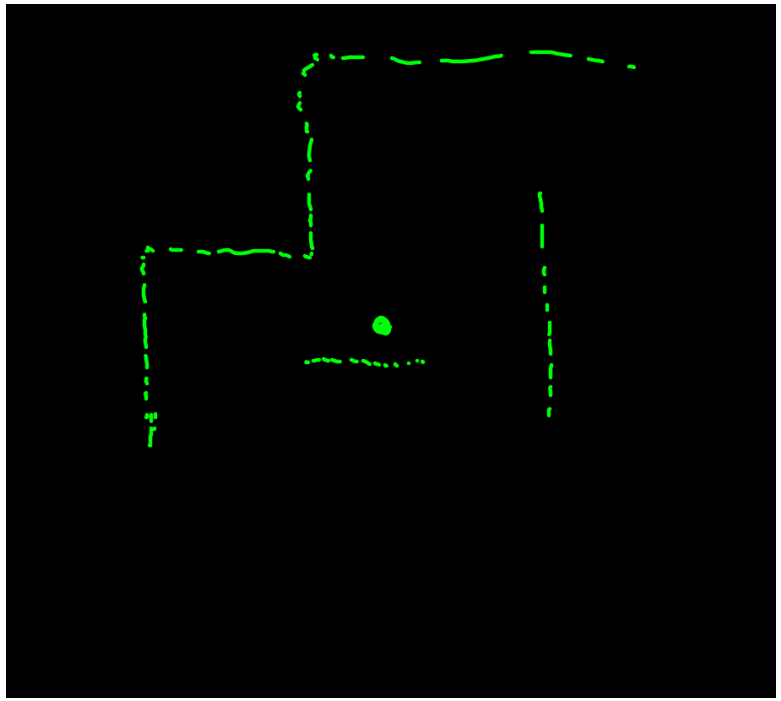


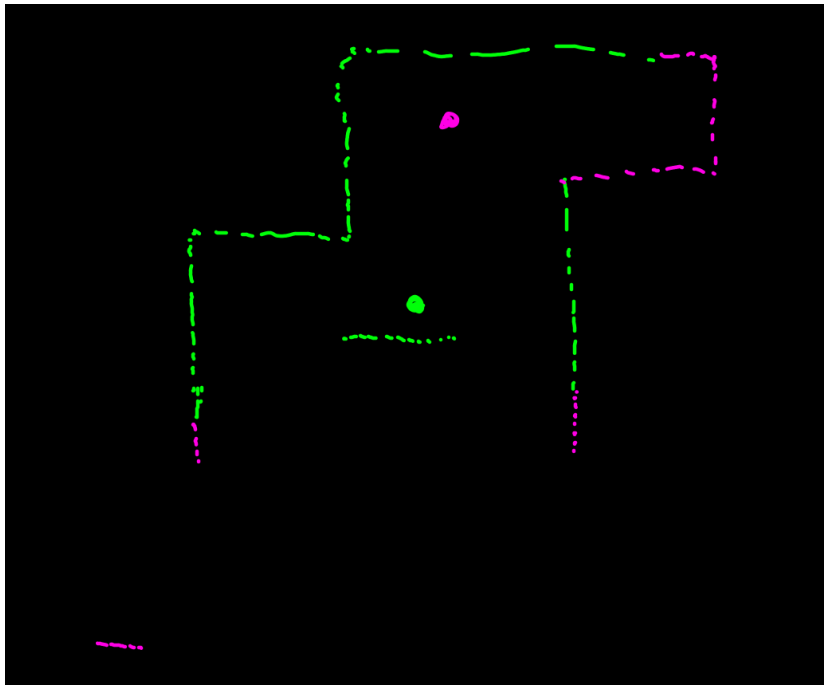


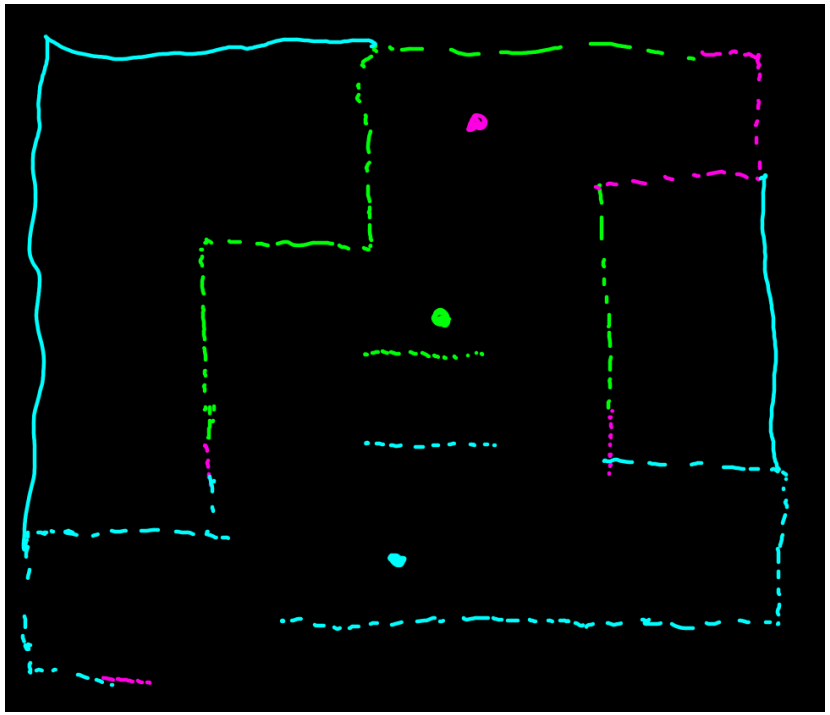
Done



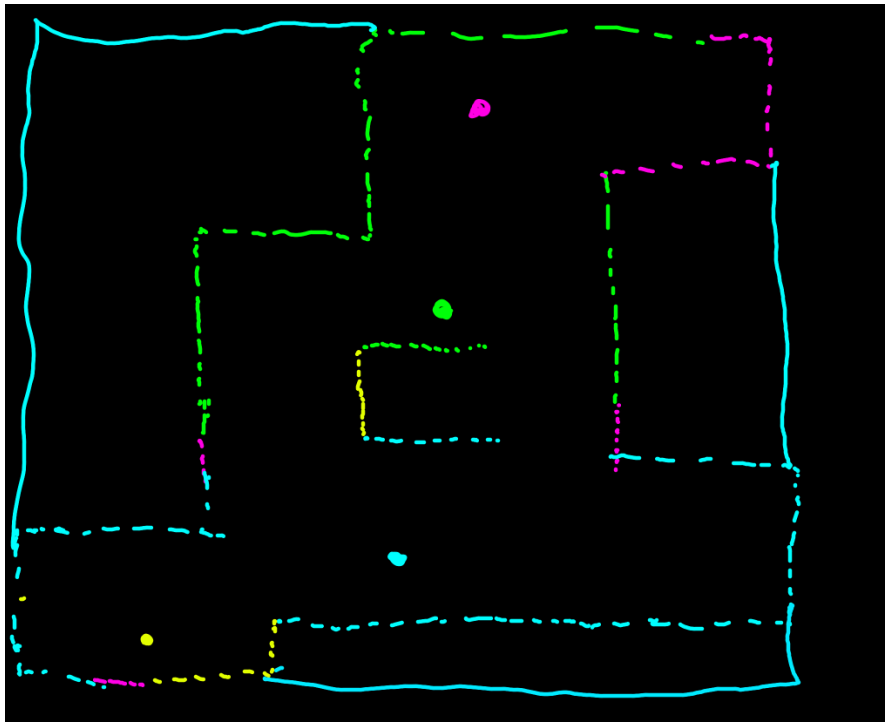
- This map took just 6 scans despite having multiple blockades. A normal map just needs 2-5 scans. Here is the original map provided in question

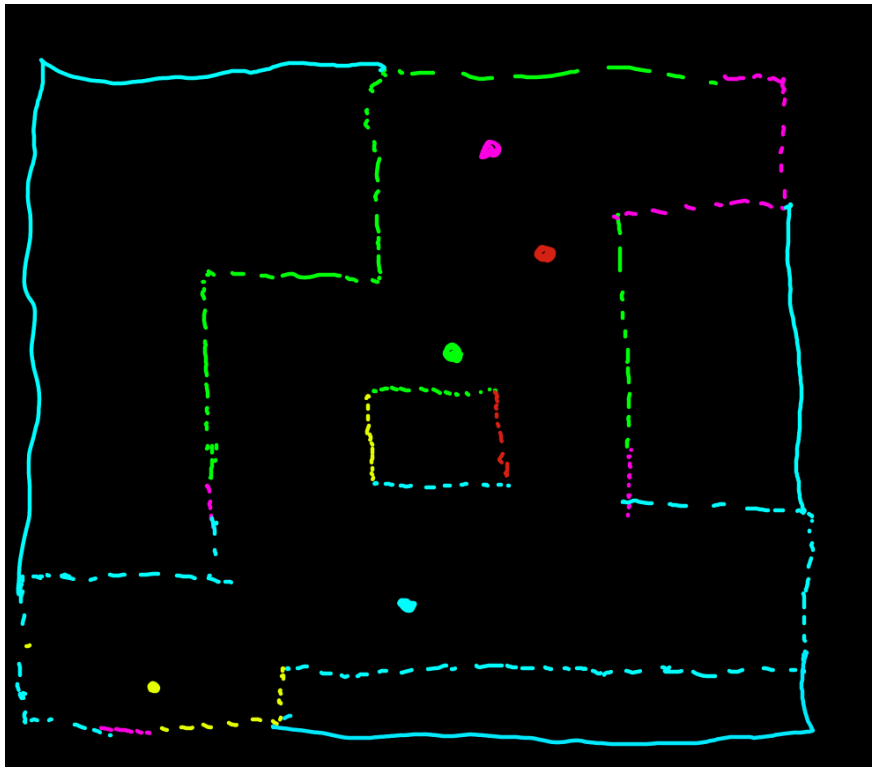






Total: 5 scans





Problems

- In some situations, instead of a boundary we may need to use another normal line to calculate the right triangle
- A lot of the times, the scan may be useless and may not be the best point to scan next, but overall, this algorithm is very efficient.