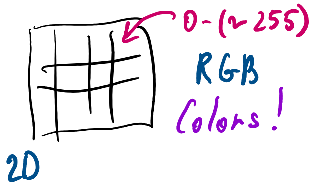
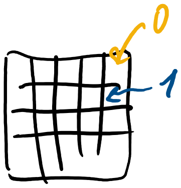
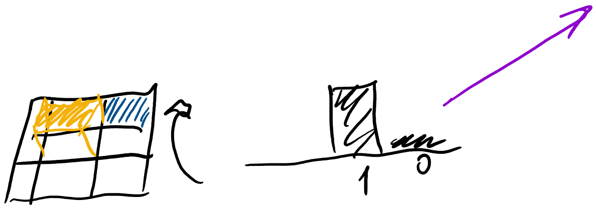
Sunday, November 10, 2024 7:13 PM

1)

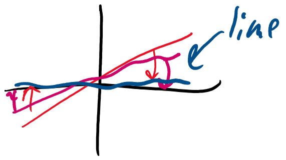


→ Open white/black



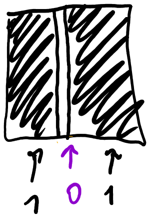


2) HT
~Hought Transform
VC(line detection)



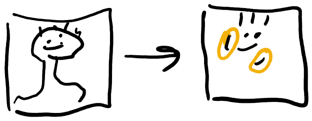
②

LSD
Line-Segment-Detection

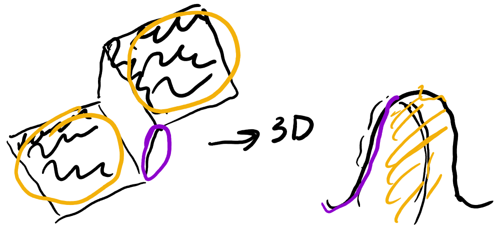


- Contrast
- Blure
- Gaussean Filters
  1st/2nd order

Edge Detection
~Sobel operator

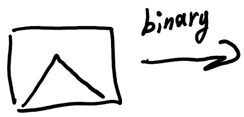


③



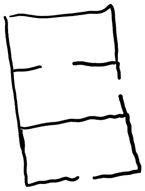
①

3) - Get the pic
- White/Black (Avoid RGB)
- print?



```
0 0 0 0 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1
```

1-edge, 0-no edge



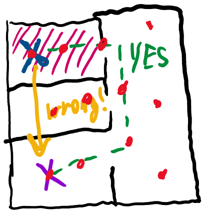
→ matrix?
1-wall
0-no wall

4) Node?

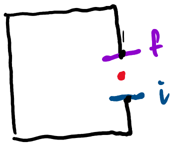
- Do BFS without hitting the walls.
- [illegible] following near 0's &

4) Node ?

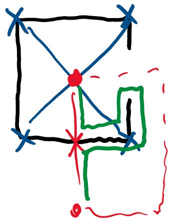
- Do BFS without hitting the walls.
- Connect nodes by following near 0's & avoiding 1's.



middle exit node ☹

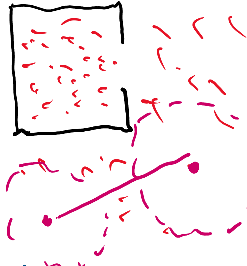


room center ☹



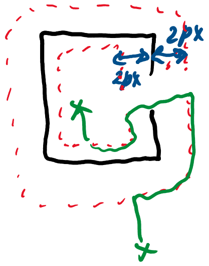
Monte Carlo

A lot nodes

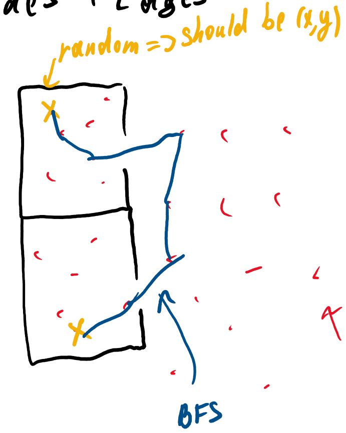


cut every 20 nodo ☺

1) cut every 20 nodes ☺
2) cut within the 5px radius ☺
3) random near the walls ☺



5) Nodes + Edges



Node = (x,y) in px

Edge = 0's the shortest

↳ BFS vs DFS

↑ good ☺

↳ too long to go in depth!

~random MC
- no radius
- every '0' is a node
  - cut every 20 nodes

floor.ph (WALLS, NODES DETECTION)
- the best walls & nodes
- return nodes (x,y)!

nodes (x,y)!

floor And Matrix.py

- generate the matrix (0's & 1's)
  - 0- white (no wall)
  - 1- green (wall)
- picks 2 random 0's (red nodes)
- runs BFS through 0's without hitting 1's.

merge.py

- get pic
- matrix (0 - no wall, 1- wall)
- pick random two 0's
- run BFS, avoid walls (1's)

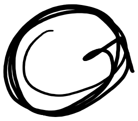
1) open pic
2) matrix
3) walls / nodes

Output pic: green red blue

Output data: red node (x, y) px

Keep for future
4) BFS (follow 0's, avoid 1's)
the shortest path

10,000 pic →



Input: 10,000 pics
Output: Nodes (x, y) → BFS

BFS
i (x,y) ; f(x,y)

→ blue path → connect with A* Albin

Impove GPT, a bunch of loops
SMAPTEP pls!!! ☺