A quick guide of the general planner

---

v1.0.1 Update:

- Now, the C++ planner can be run in the python notebook. No need to copy the map and agent files around.
- The visualization code will now load the map and agent files that are the input to the C++ planner and visualize the planned paths.

The work is still ongoing. This is a quick guide about how to use the tool.

Required tools:

- C++: the planner is a CMake project.
- Python jupyter notebook or Google colab (an online python notebook environment)

How to use:

1. Generate maps and toio start and goal locations. This results in two files: a map file and an agent file.
    - The **map** file is a design of the "roads" in the planning space.

    Example map file:

    ```
    1000,1000,625,1200
    0,20.0,20.0
    1,20.0,60.0
    2,20.0,100.0
    3,20.0,140.0
    4,20.0,180.0
    5,20.0,220.0
    6,20.0,260.0
    7,20.0,300.0
    8,20.0,340.0
    9,20.0,380.0
    ...
    623,980.0,940.0
    624,980.0,980.0
    0,1
    0,25
    1,2
    1,26
    2,3
    2,27
    3,4
    3,28
    4,5

    ....
    ```

The first line states: the x size of the planning space (1000), the y size of the planning space (1000), the number of vertices (waypoints that a toio can access) (625 vertices), the number of edges (1200).

Then the followed lines are the vertices (aka **waypoints**): vertex_id, x_coordinate, y_coordinate.

You can set the coordinates freely. But make sure that the waypoints are actually not blocked by any objects (e.g., walls).

Then the edge information lines: <vertex_id>, <connected_vertex_id>

For example, "0, 25" means the vertex with id 0 is connected to the vertex id 25.

- The **agent** file states the start and the goal locations of toios:

For example,

```
20
0,0,217
1,1,216
2,2,215
3,3,214
4,4,213
5,5,212
6,6,240
7,7,265
8,8,290
9,9,315
10,10,340
11,11,341
12,12,342
13,13,339
14,14,338
15,15,337
16,16,336
17,17,343
18,18,344
19,19,335
```

The first line: <number_of_toios>

The followed lines: <toio_id>, <start_vertex_id>, <goal_vertex_id>

2. These two files are the input to the C++ planner.
    - Compile the C++ code.
    - Run the code with these program arguments:

      ```
      ./general_planner <map_file_path> <agent_file_path>
      <suboptimality_factor> <paths_output_path>
      ```

      <map_file_path>: the location of the map file.

      <agent_file_path>: the location of the agent file.

      <suboptimality_factor>: the parameter for focal search. You can simply put 5 in there. It trades the solution cost (the sum of path lengths) for runtime.

      <paths_output_path>: where you want to output the path file.

3. Get the path file. The path file states the movements of toios in coordinates.

```
20,20,0 60,20,0 100,20,0 140,20,0 180,20,0 220,20,0 260,20,0 260,60,0
260,100,0 260,140,0 260,180,0 260,220,0 260,260,0 260,300,0 260,340,0
260,380,0 260,420,0 260,460,0 260,500,0 260,540,0 260,580,0 260,620,0
260,660,0 300,660,0 300,700,0 340,700,0
20,60,0 60,60,0 100,60,0 140,60,0 180,60,0 220,60,0 260,60,0 260,100,0
260,140,0 260,180,0 260,220,0 260,260,0 260,300,0 260,340,0 260,380,0
260,420,0 260,460,0 260,500,0 260,540,0 260,580,0 260,620,0 300,620,0
300,660,0 340,660,0
20,100,0 60,100,0 100,100,0 140,100,0 180,100,0 220,100,0 260,100,0
260,140,0 260,180,0 260,220,0 260,260,0 260,300,0 260,340,0 260,380,0
260,420,0 260,460,0 260,500,0 260,540,0 260,580,0 300,580,0 300,620,0
340,620,0
20,140,0 60,140,0 100,140,0 140,140,0 180,140,0 220,140,0 260,140,0
260,180,0 260,220,0 260,260,0 260,300,0 260,340,0 260,380,0 260,420,0
260,460,0 260,500,0 260,540,0 300,540,0 300,580,0 340,580,0
20,180,0 60,180,0 100,180,0 140,180,0 180,180,0 220,180,0 260,180,0
260,220,0 260,260,0 260,300,0 260,340,0 260,380,0 260,420,0 260,460,0
260,500,0 300,500,0 300,540,0 340,540,0
20,220,0 60,220,0 100,220,0 140,220,0 180,220,0 220,220,0 260,220,0
300,220,0 340,220,0 340,260,0 340,300,0 340,340,0 340,380,0 340,420,0
340,460,0 340,500,0
20,260,0 60,260,0 100,260,0 140,260,0 180,260,0 220,260,0 260,260,0
300,260,0 340,260,0 380,260,0 380,300,0 380,340,0 380,380,0 380,420,0
380,460,0 380,500,0 380,540,0 380,580,0 380,620,0
20,300,0 60,300,0 100,300,0 140,300,0 180,300,0 220,300,0 260,300,0
300,300,0 340,300,0 380,300,0 420,300,0 420,340,0 420,380,0 420,420,0
420,460,0 420,500,0 420,540,0 420,580,0 420,620,0
20,340,0 60,340,0 100,340,0 140,340,0 180,340,0 220,340,0 260,340,0
300,340,0 300,380,0 300,420,0 340,420,0 380,420,0 380,460,0 380,500,0
380,540,0 380,580,0 420,580,0 460,580,0 460,620,0
20,380,0 60,380,0 100,380,0 140,380,0 180,380,0 220,380,0 260,380,0
300,380,0 300,420,0 300,460,0 340,460,0 380,460,0 420,460,0 420,500,0
420,540,0 460,540,0 500,540,0 500,580,0 500,620,0
20,420,0 60,420,0 100,420,0 140,420,0 180,420,0 220,420,0 260,420,0
300,420,0 300,460,0 300,500,0 340,500,0 380,500,0 420,500,0 420,540,0
420,580,0 460,580,0 500,580,0 500,620,0 540,620,0
20,460,0 60,460,0 100,460,0 140,460,0 180,460,0 220,460,0 260,460,0
300,460,0 300,500,0 300,540,0 340,540,0 380,540,0 380,580,0 420,580,0
460,580,0 500,580,0 500,620,0 540,620,0 540,660,0
20,500,0 60,500,0 100,500,0 140,500,0 180,500,0 220,500,0 260,500,0
300,500,0 340,500,0 380,500,0 420,500,0 460,500,0 500,500,0 500,540,0
500,580,0 500,620,0 540,620,0 540,660,0 540,700,0
20,540,0 60,540,0 100,540,0 140,540,0 180,540,0 220,540,0 260,540,0
300,540,0 340,540,0 380,540,0 420,540,0 460,540,0 500,540,0 500,580,0
500,620,0 540,620,0 540,580,0
20,580,0 60,580,0 100,580,0 140,580,0 180,580,0 220,580,0 260,580,0
300,580,0 340,580,0 380,580,0 420,580,0 460,580,0 500,580,0 540,580,0
540,540,0
20,620,0 60,620,0 100,620,0 140,620,0 180,620,0 220,620,0 260,620,0
300,620,0 340,620,0 380,620,0 420,620,0 460,620,0 500,620,0 540,620,0
540,580,0 580,580,0 580,540,0 580,500,0 540,500,0
20,660,0 60,660,0 100,660,0 140,660,0 180,660,0 220,660,0 260,660,0
300,660,0 340,660,0 380,660,0 420,660,0 460,660,0 460,620,0 460,580,0
460,540,0 500,540,0 500,500,0 540,500,0 540,460,0
```

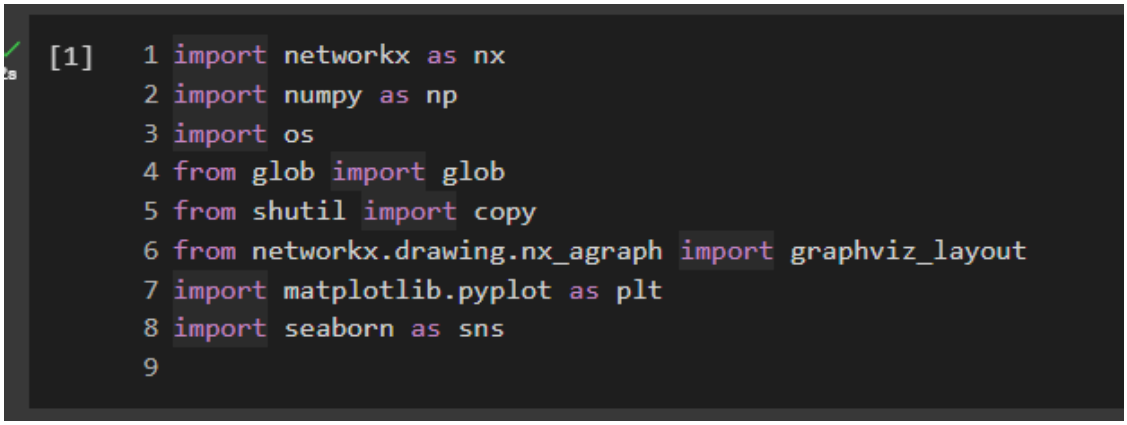
```
20,700,0 60,700,0 100,700,0 140,700,0 180,700,0 220,700,0 260,700,0
300,700,0 340,700,0 380,700,0 420,700,0 460,700,0 500,700,0 540,700,0
540,740,0
20,740,0 60,740,0 100,740,0 140,740,0 180,740,0 220,740,0 260,740,0
300,740,0 340,740,0 380,740,0 420,740,0 460,740,0 500,740,0 540,740,0
540,780,0
20,780,0 60,780,0 100,780,0 140,780,0 180,780,0 220,780,0 260,780,0
300,780,0 300,740,0 300,700,0 340,700,0 380,700,0 420,700,0 460,700,0
460,660,0 460,620,0 460,580,0 460,540,0 500,540,0 500,500,0 500,460,0
500,420,0 540,420,0
```

Each line is the movement of a toio in coordinates (x,y,z). The z-axis movement is not implemented at this moment.

The toio stays at the goal location once it reaches the location.

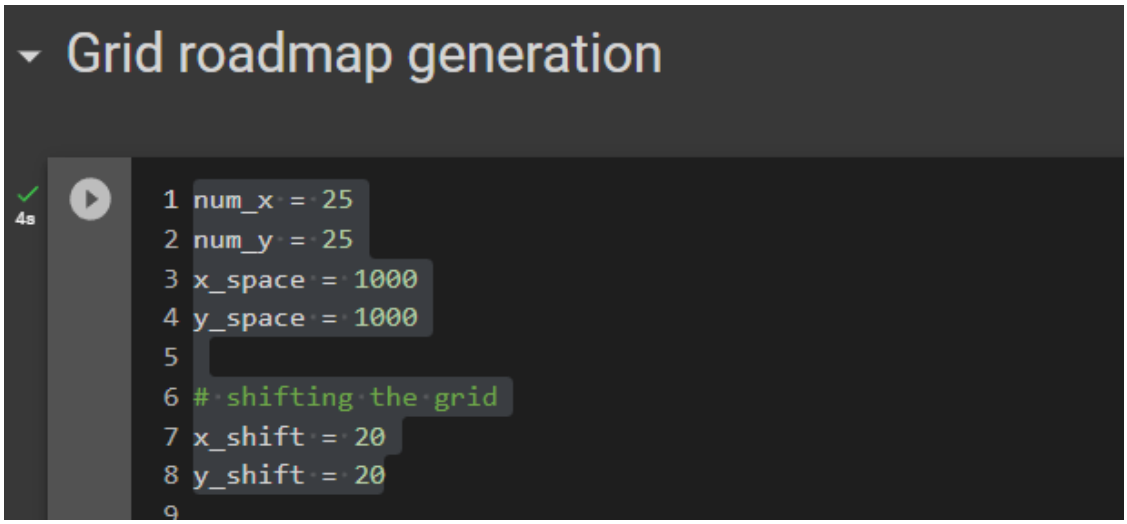
Walkthrough of an example:

1. Open the python notebook ("map_generator_and_visulization.ipynb")
2. Run the library import block



```
[1]  1 import networkx as nx
     2 import numpy as np
     3 import os
     4 from glob import glob
     5 from shutil import copy
     6 from networkx.drawing.nx_agraph import graphviz_layout
     7 import matplotlib.pyplot as plt
     8 import seaborn as sns
     9
```

3. Then move on to the code for generating a grid general map.



```
Grid roadmap generation

4s
1 num_x = 25
2 num_y = 25
3 x_space = 1000
4 y_space = 1000
5
6 # shifting the grid
7 x_shift = 20
8 y_shift = 20
9
```

You only need to change these parameters.

num_x: how many vertices are in the x-axis

num_y: how many vertices are in the y-axis

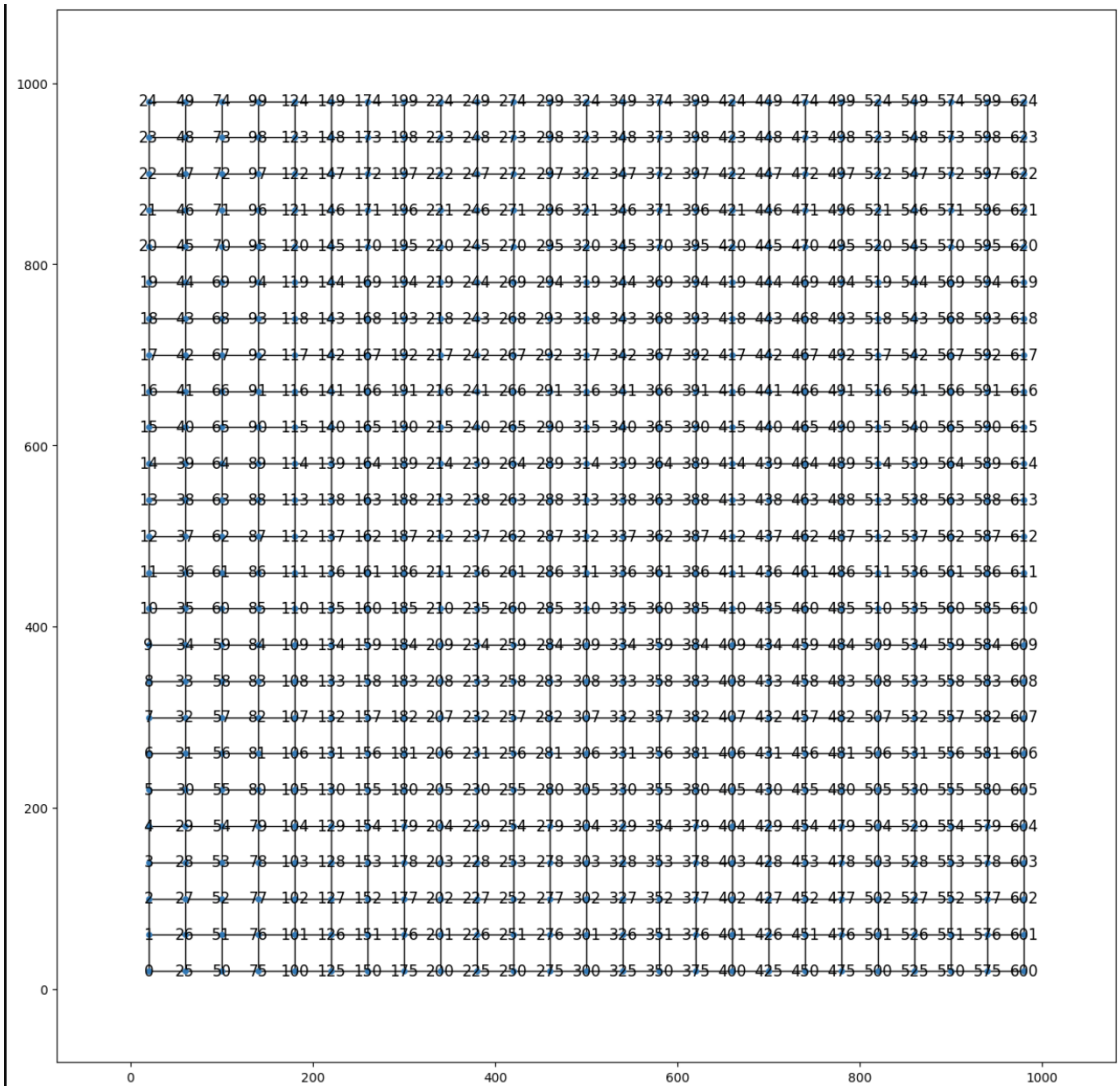
x_space: the x size of the map

y_space: the y size of the map

x_shift: it shifts the x coordinates of all vertices.

y_shift: it shifts the y coordinates of all vertices.

Run the code, and you will see this picture output.



This is the grid "roadmap" generated by the code. Each vertex is the waypoint where the toio can be on, and each vertex is labelled with its vertex id.

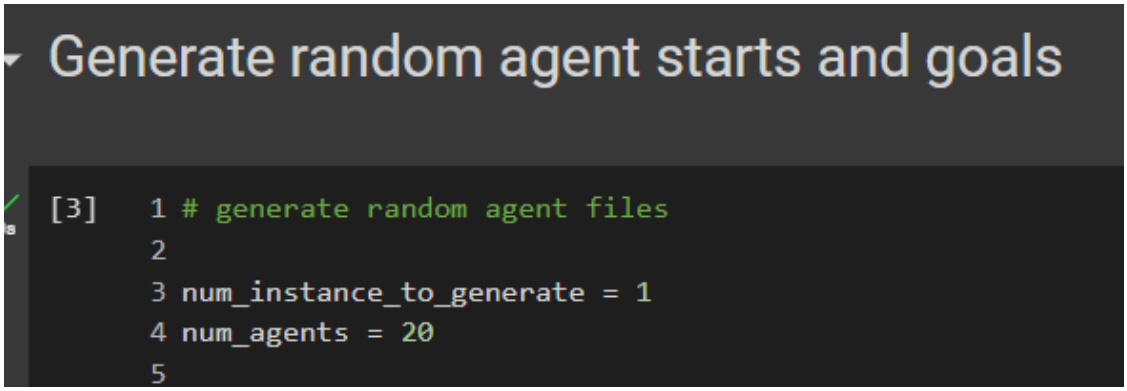
The display of the vertex id can be helpful, if you want to write down the start and goal locations of toios mannually.

After running this block of code, it generates a map file "grid_map_x_x_x.txt" in the "toio_map_example" folder.

This is the map file that you need for using the C++ planner.

4. Generate an agent file.

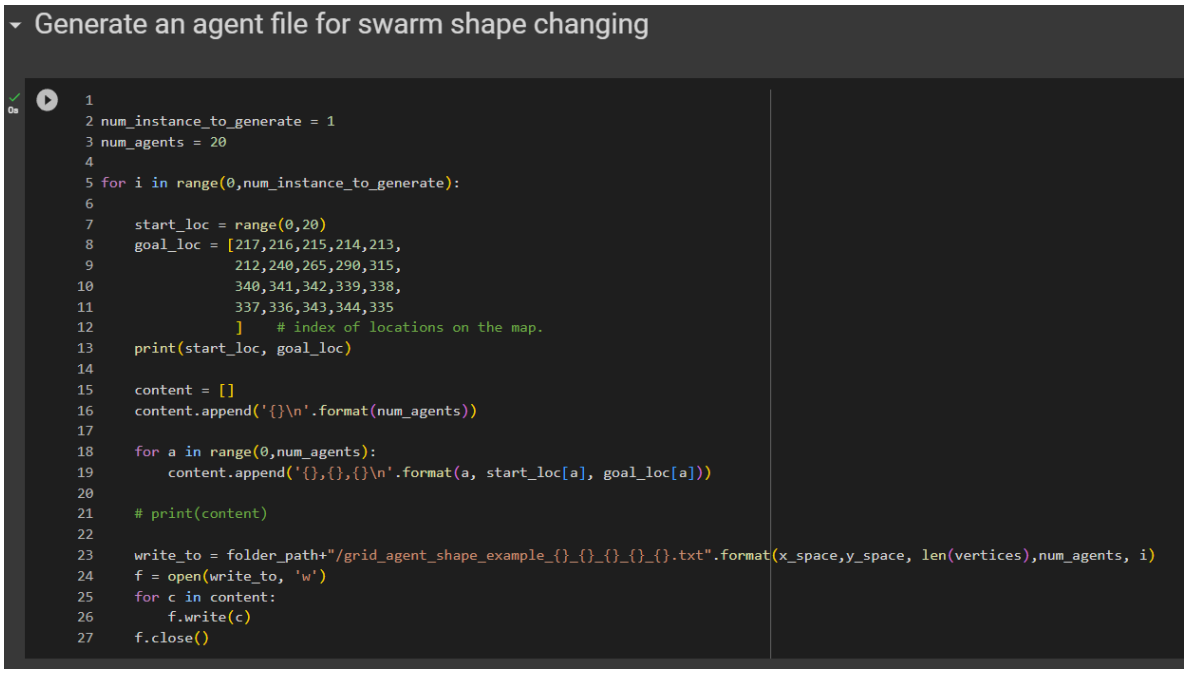
   This block of code generates one agent file with random start and goal locations.



num_instance_to_generate: how many agent files will be generated.

num_agents: how many toios are used.

You can use this block of code to generate a customized agent file.



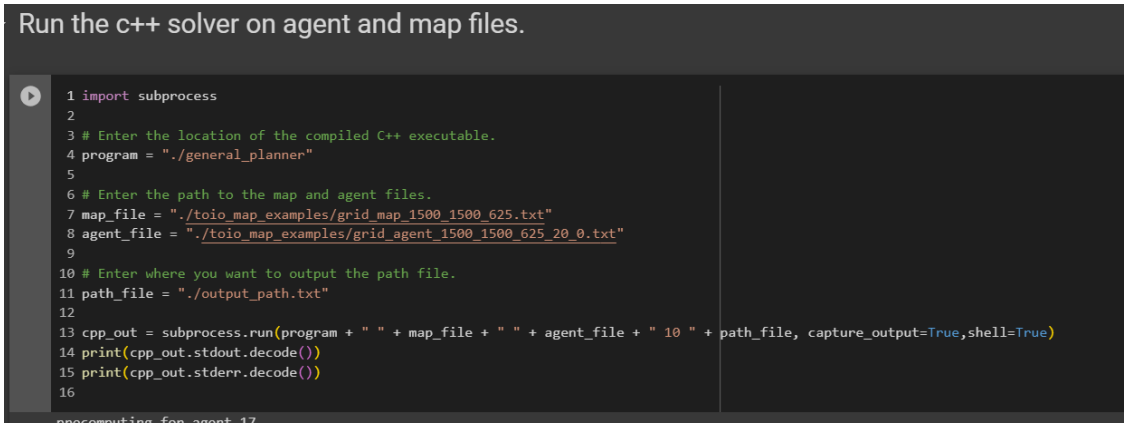
start_loc (line 7): the start vertex id for each agent.

goal_loc (line 8): the goal vertex id for each agent.

You can look at the map image generated in the previous step to pick the start and goal locations.

write_to (line 23): the path to the agent file. In this case, you will find an agent file in the "toio_map_examples" folder.

5. Compile the C++ planner and go to this block.



program: enter the path to the compiled C++ planner executable.

map_file: which map file you want to load for planning.

agent_file: which agent file you want to load for planning.

path_file: where to output the result paths.

You can run the planner by just running this block of code.

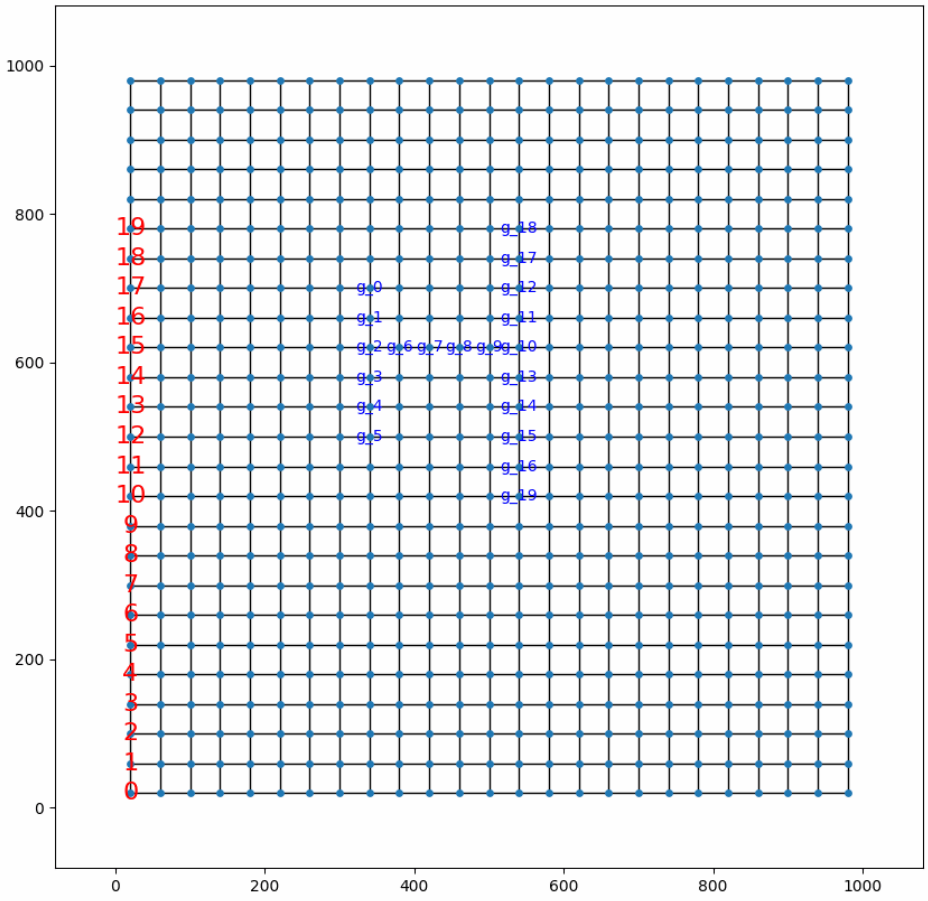
6. Then just run the visualization code.

   Now the visualization code will visualize the map and the movements that are just planned by the C++ planner.

   The code reads the paths that are stored in "map_file", "agent_file", and "path_file" in the running C++ planner code block (the inputs to the planner and the output from the planner.).

   Example movement:

   If this git doesn't move in this document, check toio_movement.gif in the folder.


1000
800
600
400
200
0
0
200
400
600
800
1000
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
g_18
g_17
g_12
g_11
g_10
g_13
g_14
g_15
g_16
g_19
g_0
g_1
g_2
g_6
g_7
g_8
g_9
g_3
g_4
g_5