RWA2

ENPM809Y: Introductory Robot Programming

Author: Kiran S Patil
(119398364)

Teammates: Rishikesh Jadhav
(119256534)

Nishant Pandey
(119247556)

Contents:

1. Package

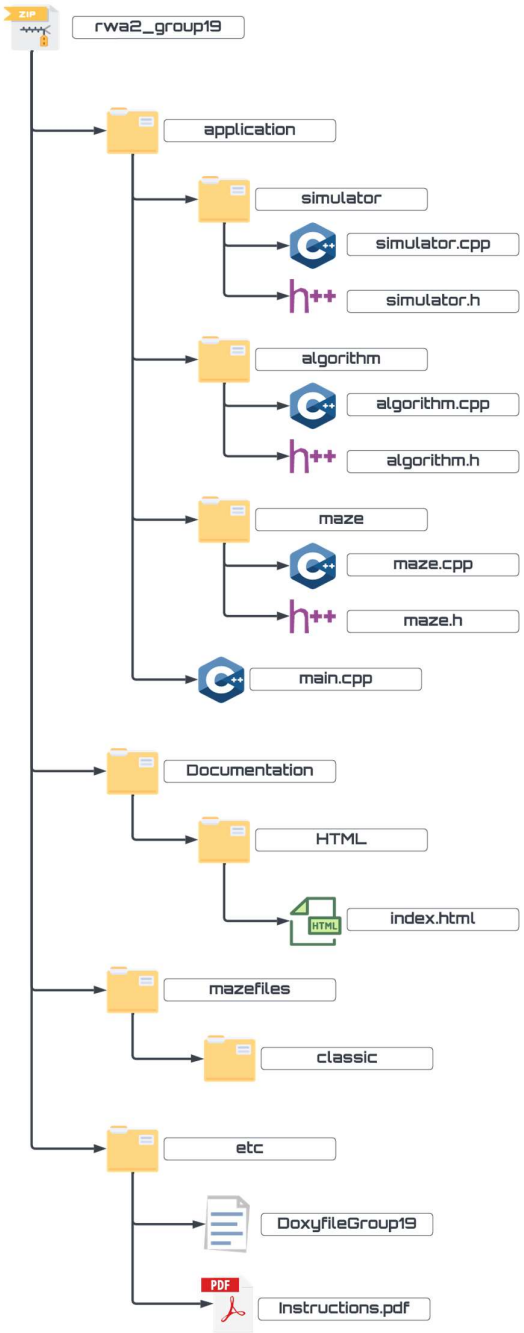

Fig. 1: Package structure for RWA2

2. Simulator Setup

- First include a mazefile in Config->Maze.
  Mazefiles can be found under */rwa2_group19/mazefiles/classic*

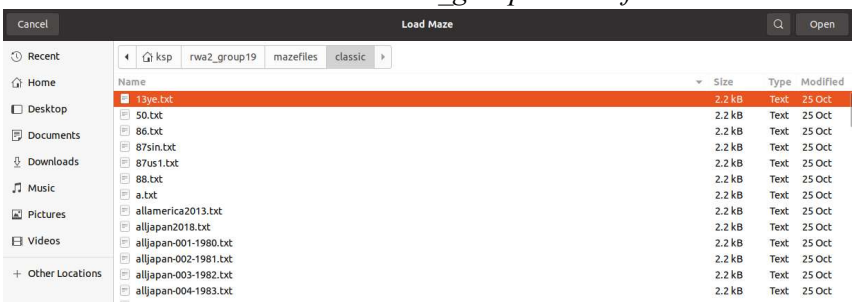

Fig. 2: Mazefiles

- Next Setup build commands in Config->Mouse.
  - Navigate to application folder; can be found at */rwa2_group19/application*
  - Build Command: need to include all .cpp files
    g++ -std=c++17 –g main.cpp algorithm/algorithm.cpp maze/maze.cpp simulator/simulator.cpp –o main
    Run Command: **./main**

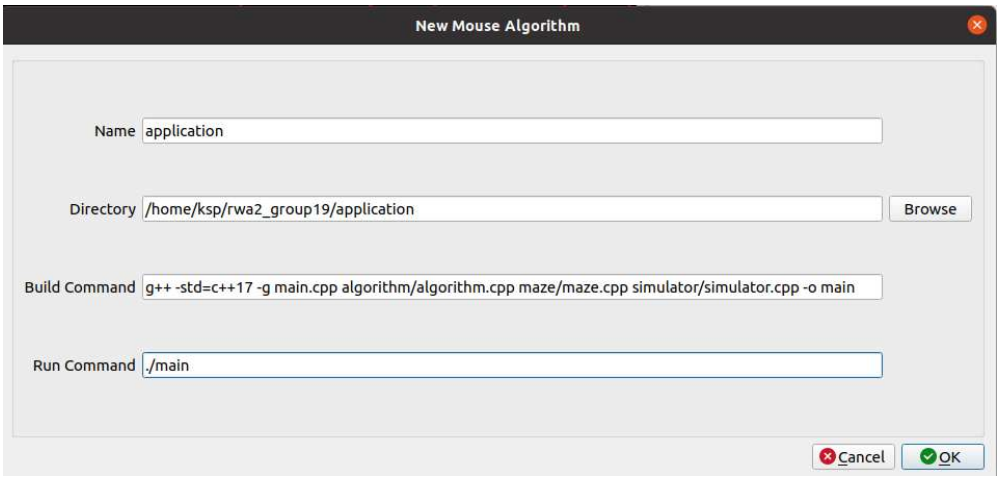

Fig. 3: commands to be set in Config->Mouse

3. Run

- By default Left Wall Follower algorithm is been implemented

➔ Start of Run: methods from SetMaze are used to set boundary walls, cell coordinates and generate a random goal (represented in a red cell with text ‘G’, cell (13,0) in the below fig. Note: Left wall following algorithm is displayed in Run Output window.

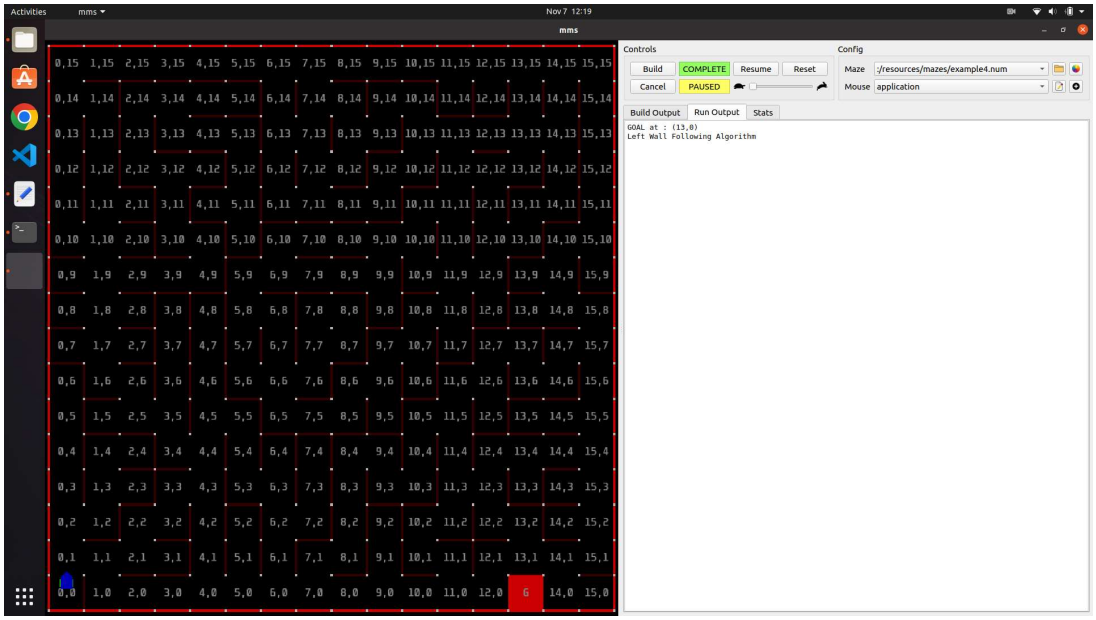

Fig. 3.1: Start of Run

➔ Detecting dead ends: once the mouse finds itself in a cell which has all three walls i.e. left, front and right wall, this cell color is changed (dark red) & also its left and front wall is set. Note: also the path followed by the mouse is shown in cyan color.

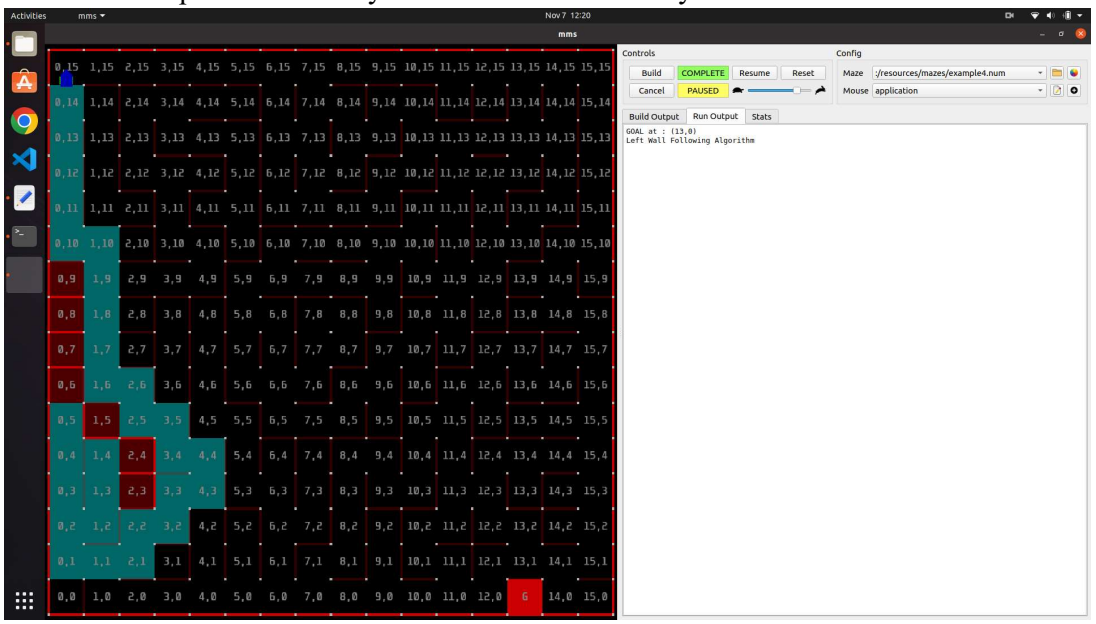

Fig. 3.2: Detecting dead ends

➔ Goal Reached: Once the goal is reached, the optimum path to reach back home is calculated (path shown in green) and the excess path (shown in dark red) is excluded.

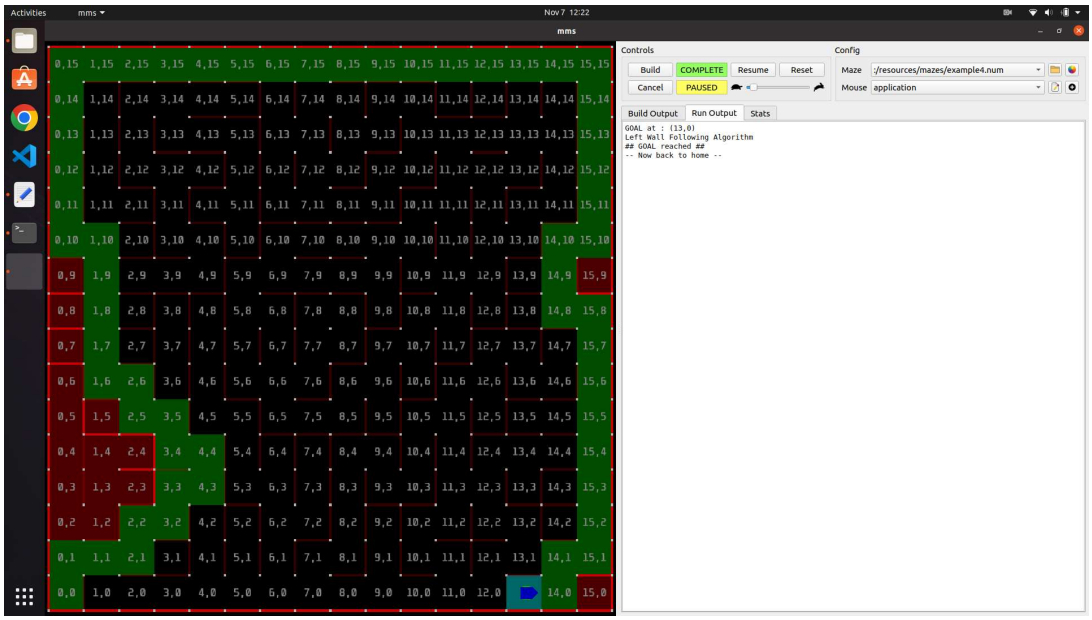

Fig. 3.3: Goal Reached

➔ Back Home: Next the mouse follows the optimum path to reach back home.
Note: The path followed by the mouse is shown in cyan color.

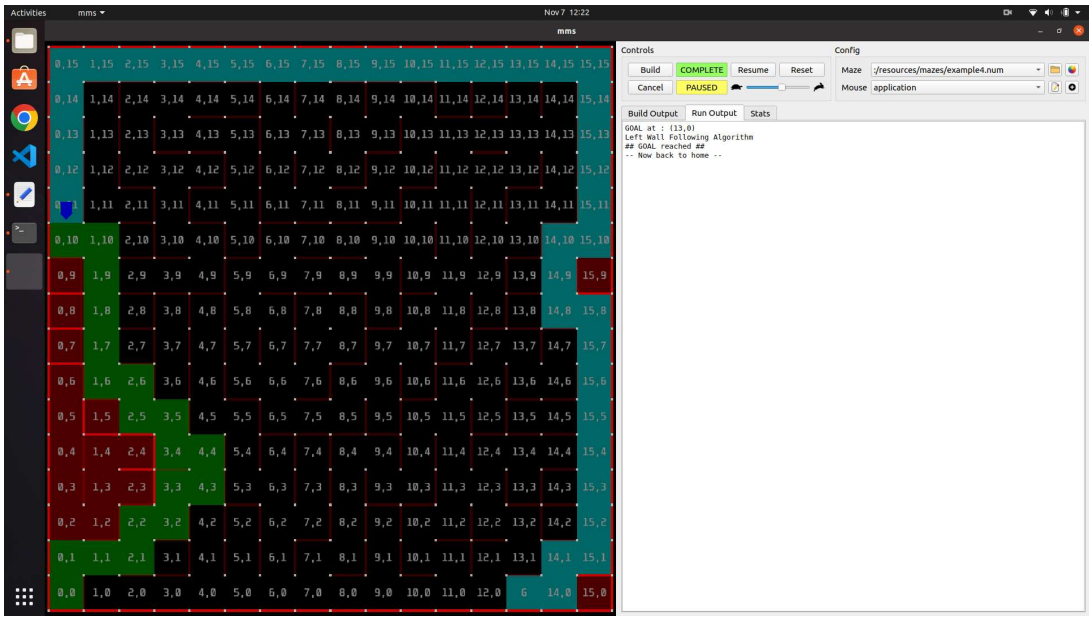

Fig. 3.4: Back Home

➔ End of Run: Once the mouse reaches back home i.e. (0,0) concludes the run.

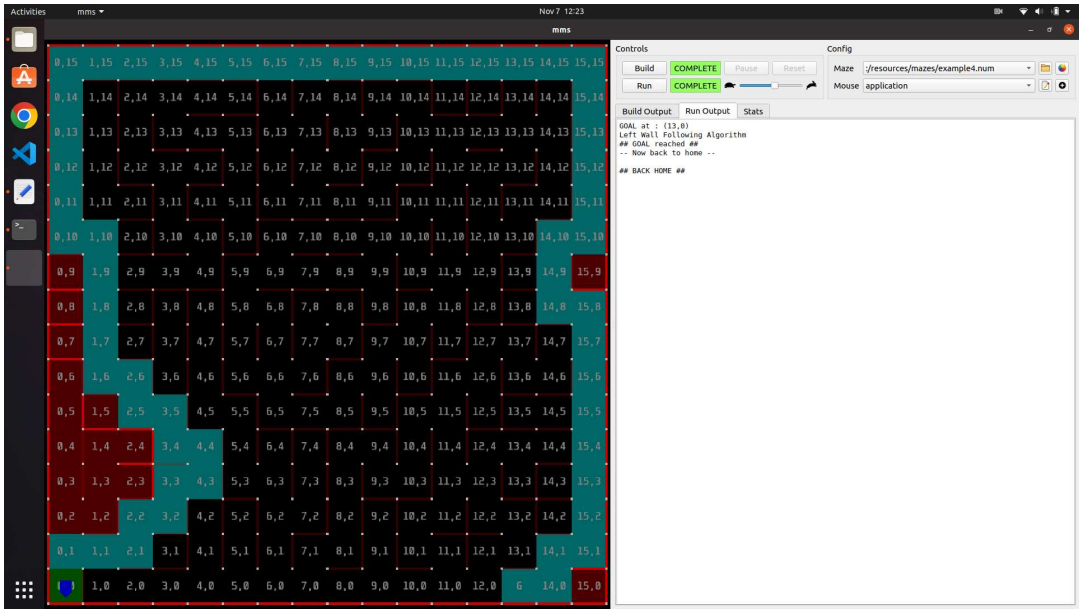

Fig. 3.5: End of Run

4. Switch to Right Wall Following Algorithm:

To switch to right wall following algorithm instead of left wall following algorithm we need to comment the line which is instantiating method m_left_wall_follower, and uncomment the line which instantiates the method m_right_wall_follower.

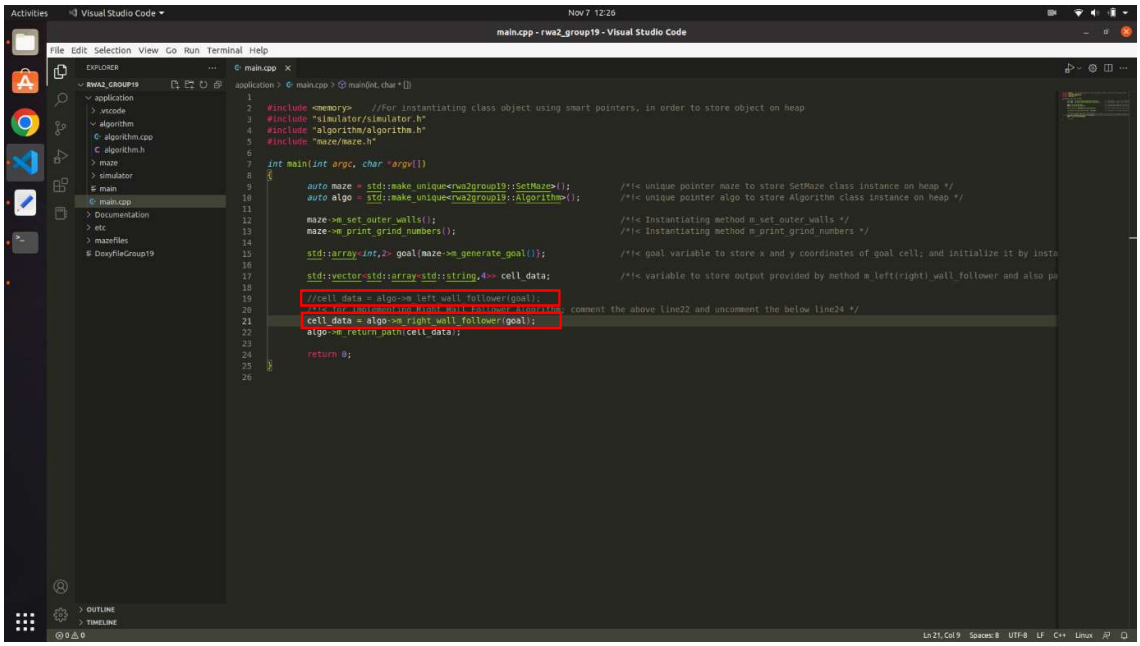

Fig. 4.1: LWF to RWF

Note: Right wall following algorithm is displayed in Run Output window.

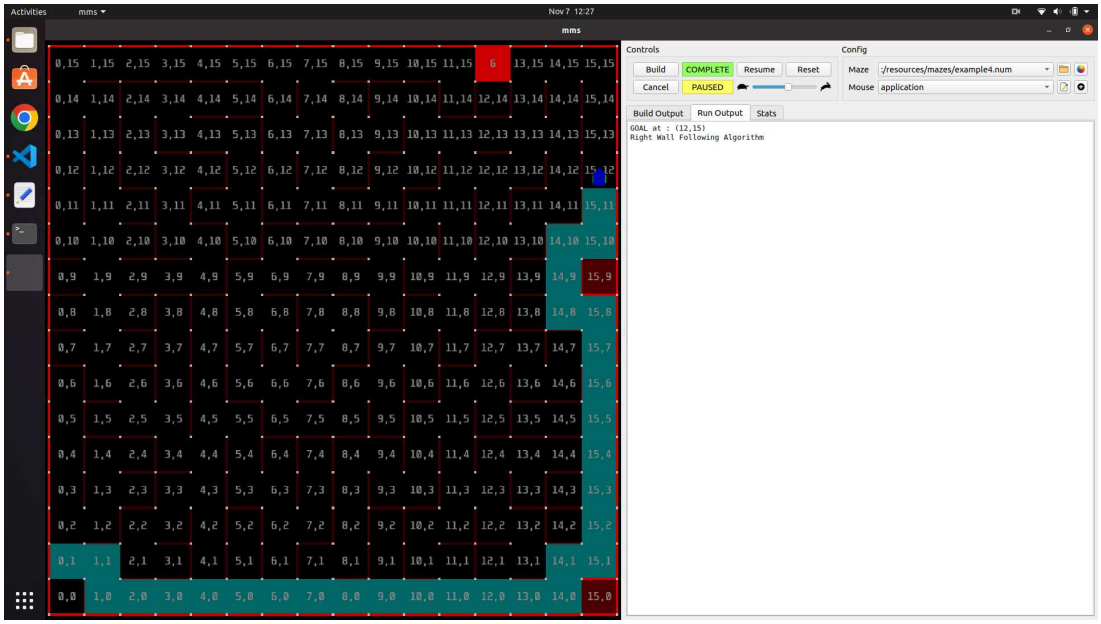

Fig. 4.2: Right wall follower run

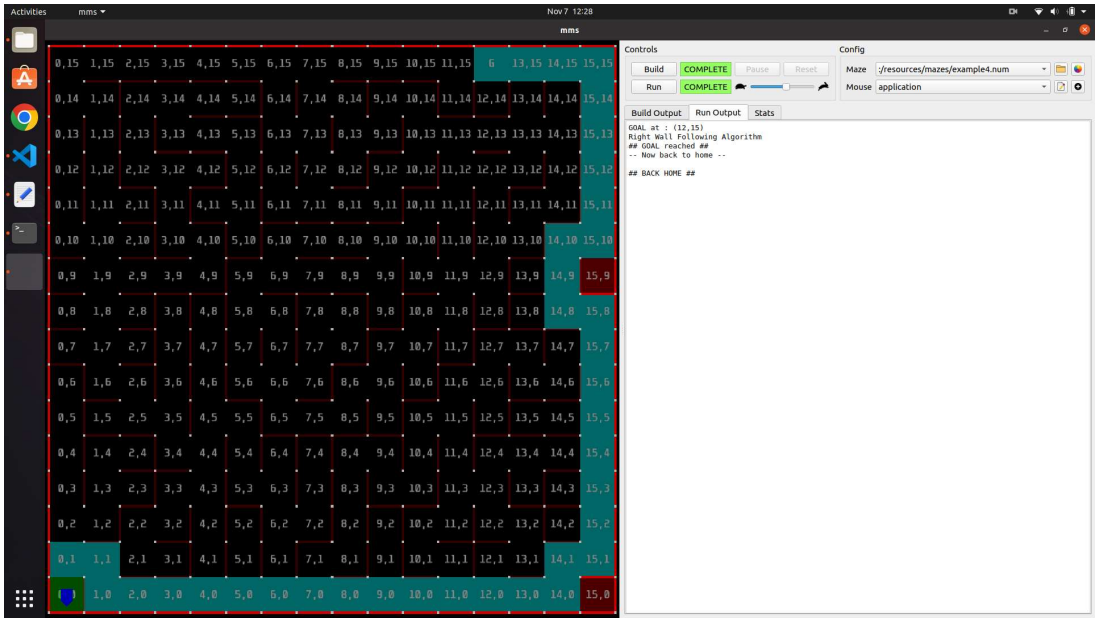

Fig. 4.2: End of right wall follower run

5. Documentation:

Doxygen generated index.html file can be found at */rwa2_group19/Documentation/HTML*

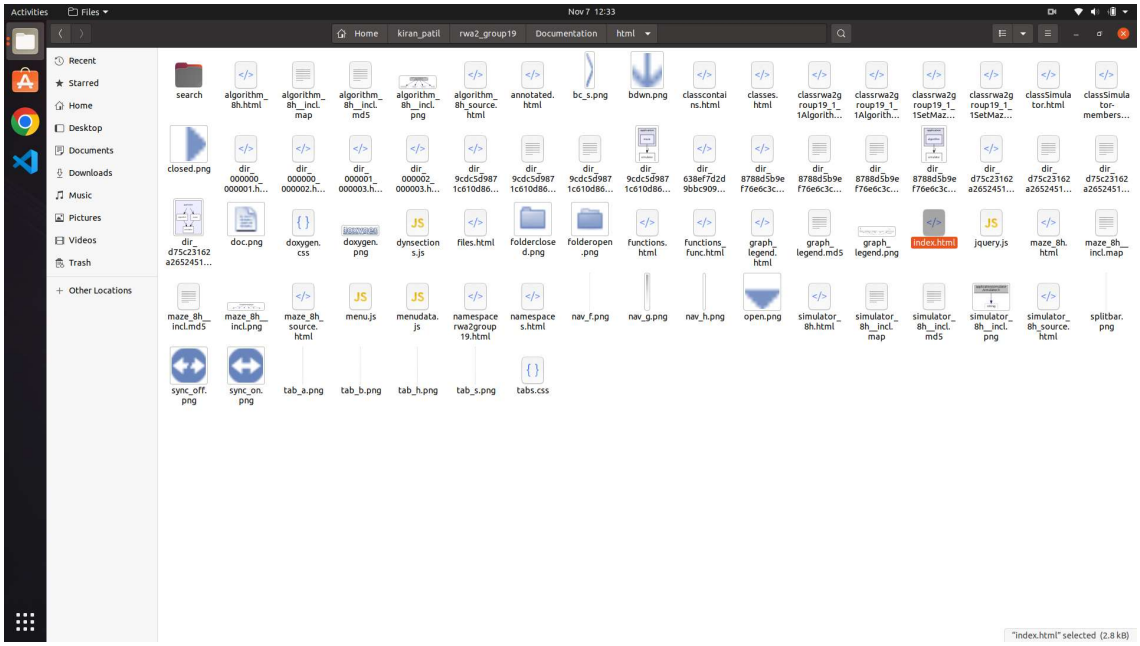

Fig. 5.1: Navigating to index.html

index.html opens up a webpage (sample shown below), which provides documentation. This includes all the sections individually like namespace and classes, and also furthermore details about this sections like class methods.

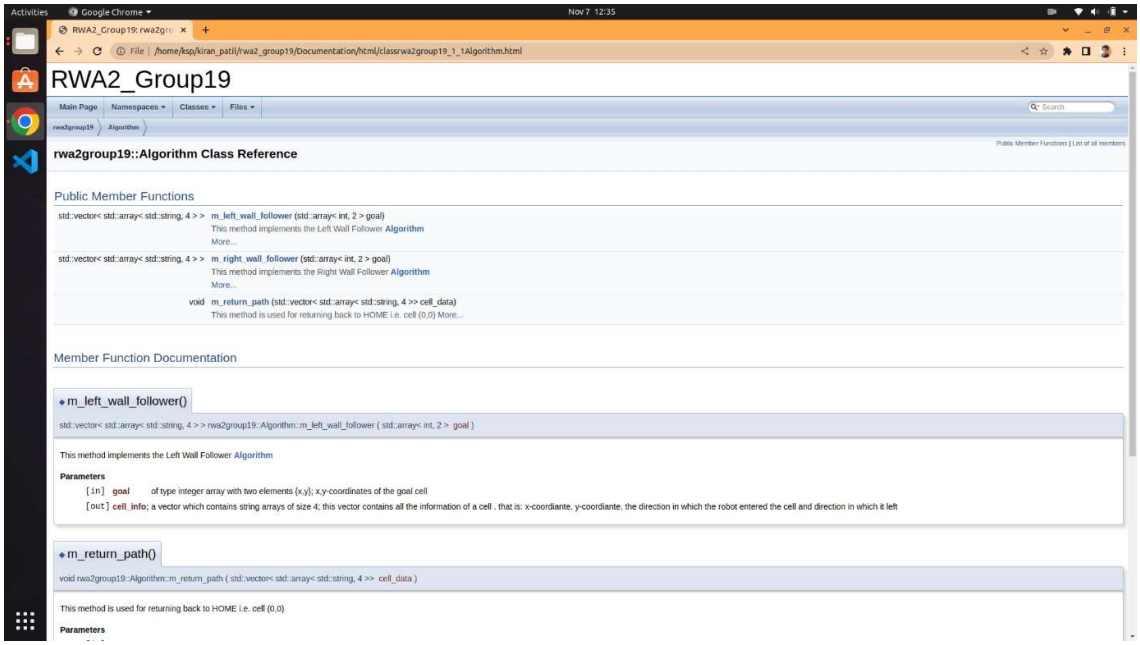

Fig. 5.2: Doxygen generated documentation

6. Bugs

The mouse fails to reach the goal if the random goal generated unfortunately is an *inf* cell, i.e. the cell is covered by walls on all the four sides.

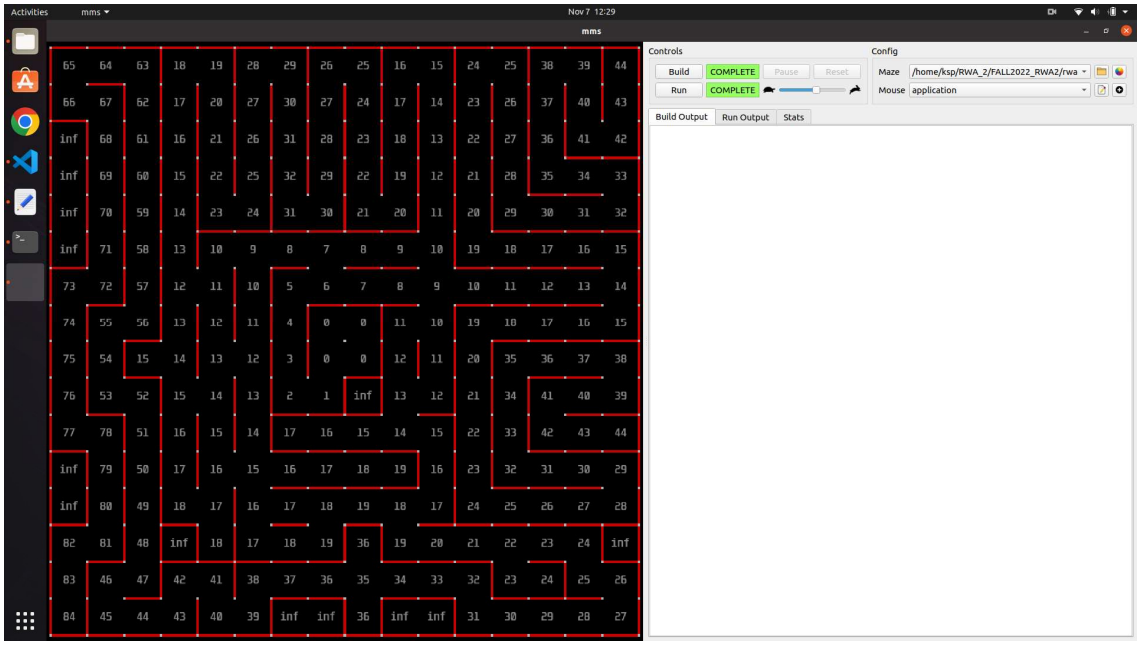

Fig. 6.1: Maze with inf cells at boundary

The below figure has a goal at cell (9, 0), which unfortunately in an inf cell. Thus the mouse can never reach its goal and continue an infinite run.

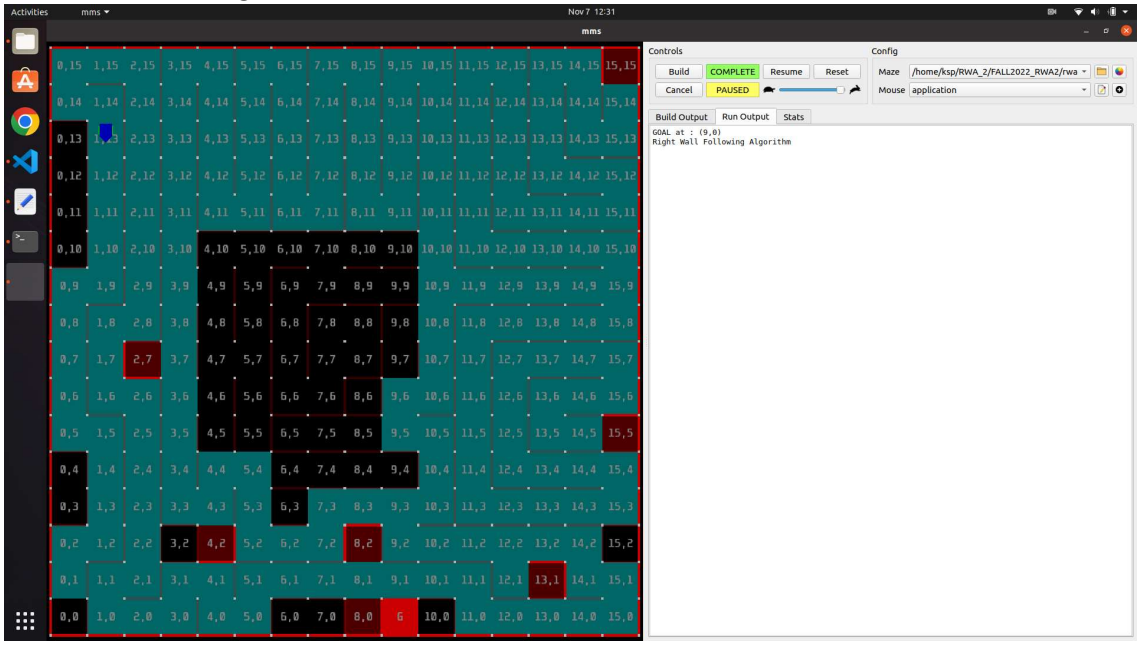

Fig. 6.2: Infinite Run