Coordinated-Motion-Planning

In this project we will take part in the competition.

The competition has two separate categories, with the two following objective functions:

1. minimize the makespan, i.e., the time until all robots have reached their destinations.
2. minimize the total distance traveled by all robots.

Class Start Game:

This class represents the game.

- start_game(eb): This function run until all the robots reached their place.

Class Extract Board:

This class represents the initialization of the board.

- extract_board(): Takes from the Json file all the information on the board.
- fill_board(): Initializes the board with the current position of the robots, the final position and the obstacles. And initializes the list of robots.

Class init frames:

This class represents the initialization of the frame.

- init_frames(eb): Moves all robots to the board frame

Class insert in spiral:

This class represents the passage of the robots to the final location .

- insert_in_spiral(eb): Runs on the original board and wherever it is the final location of a robot, moves the robot to it

Class Move Robot:

This class represents the Functions of moving robots.

- move_robot(eb, robot, robot_queue_node): Moves the robot along the path it received
- move_robot_one_step(eb, dest, robot) : Moves the robot one step
- move_robots_back(eb, robot_prev_place): Moves the robots on the list back to where they were.
- move_robot_stack(eb, robot, robot_queue_node): Moves robots that interfere with a particular robot to move to its final position.
- get_direction(p, n): Returns the quarter in which the robot is located.

Class Move Robot By Direction:

This class represents the Functions of moving robots by direction.

- move_robots_up(eb, robot): Moves robots that are in the frame up to their final position
- move_robots_down(eb, robot): Moves robots that are in the frame down to their final position
- move_robots_left(eb, robot): Moves robots that are in the frame left to their final position
- move_robots_right(eb, robot): Moves robots that are in the frame right to their final position

Class Robot Details:

This class represents details about the robots.

- find_robot_by_number(num): return the robot with the number.

Class Increase Board:

This class represents the increase board.

- row_space(number_of_robots, n): Returns the number of rows to which the board should be increased. According to what he gets.

Class shortestPath:

This class calculates the shortest path given the starting point, end point and board.

- isValid(row: int, col: int ,ROW: int, COL: int)- gets board size and point (row,col),return if the point is inside the board
- BFS(board, src: Point, dest: Point): Gets a board, a source point and an end point. Calculates the shortest path on the board considering other robots and obstacles. Returns an object quenueNode that contains a source point, the length of the shortest path and all the steps of the shortest path.

Class quenueNode:

This class has an object that is returned from the BFS function and saves the shortest path to a specific robot.

- __init__(self, current_point: Point, dist: int, path: []) – the constructor, gets current Point, the length of the shortest path and all the steps of the shortest path.

Class Robot:

This class represents the robot object.

- __init__(self, current_place:Point, end_place:Point): The constructor, gets current Point and end Point and create a Robot.

Class Point:

This class represents the Point object

- __init__(self, x: double, y: double): The constructor, gets X and Y and create a Point.
- equal(p1, p2): Gets tow Points and return if the Points is equals.