AMR REPORT
Team-02

Motion Controller:

Procedure:

Initial point: (x, y, θ) = (-6.5, -6.5, 45^o)

Test Cases:

1. Check for the combination of linear and angular motion and alignment in a straight line. Below are start position to goal position. It covers all quadrant, positive and negative dx and dy, and big and small dθ.
- -6.5, -6.5, 45^o → 3, -3, 0^o
- 3, -3, 0^o → 2, 5, 180^o
- 2, 5, 220^o →-3, 2, 110^o
2. Check the angular velocity value keeping the goal point very near
3. Plotted and validated against the velocity profile graph.

Test Cases	Moves in a Straight Line and combines linear and angular motions.	Velocity within maximum value	Follows velocity profile.
Iswariya	Yes	No (Only linear velocity)	No (Only linear velocity)
Widya	Yes	No (Only angular velocity)	No (Only linear velocity)
Janhavi	Yes	No	No

Conclusion:

All three codes are able to move the robot in a straight line. However all are fail on meeting the maximum velocity requirement, either it is linear velocity, angular velocity, or both. Therefore, for motion_controller node, it will be a combination of the three code. Some improvement will be made includes velocity limitation and angular velocity proportion to linear velocity.

Wall Follower:

Procedure:

Test Cases:

Case 1: Handles convex corners

Case 2: Handles concave corners

Case 3 : Finds Wall

Case 4: Follows the Wall

Case 5: Works for both worlds

Test Cases	Case1	Case2	Case3	Case4	Case5
Iswariya	Partial	Yes	Yes	Yes	Partial
Widya	Yes	Yes	Yes	Yes	Yes
Janhavi	No	No	Yes	No	No

Conclusion: The results for first two programs were good in comparison to the last one which has some problem in aligning to the wall.So for this code we are using combination of first two along with some improvements in handling convex corners and the doors.

Bug2:

Procedure:

Test Cases:

1. **Finding goal in simple environment**

   Test the code with goal point (1,1) keeping the bot in the First Quadrant

2. **Finding goal in complex environment**

   Test the code with goal point (-4,-4) keeping the bot in the Fourth Quadrant

3. **Unreachable goal**

   Test the code with goal point as (3,3) keeping the bot in the Second Quadrant

Test Cases	Finding goal in simple environment.	Finding goal in complex environment.	Unreachable goal	Maintaining clearance with the wall in Complex Env
Iswariya	Yes	Yes	No	Partial
Widya	Yes	Yes (Not optimal path)	No	Yes
Janhavi	Yes	No	No	Yes

Conclusion: For this results all the three were able to find the path in simple environment.But as the first one has better results in complex environments we are selecting this one along with addition of the unreachable goal condition and

changing some conditions so that the robot maintains clearance in the complex environment also.