<u>GROUP 5 DESIGN DOCUMENT</u>

1 Requirements

For this assignment, we are expected to develop and implement a software code so that Pico can autonomously solve a maze. The assignment consists of the following challenges:

1.1 Corridor Challenge

- Pico must drive through a corridor and take the first exit.
- It must not touch the wall.
- Must detect the exit on its own.
- We have total 5 minutes to complete the challenge.

1.2 Maze Challenge

- Pico must be able to solve the maze and find the exit point which is not known.
- It should detect the door and send the request command for the door to open.
- The door opens when the robot is stationary.
- We have total 7 minutes to complete the challenge.

2 Functionality

Based on the requirements mentioned above, certain functionalities of the robot at task were detected. Based on levels of complexity these can be divided into 3 main groups:

2.1 Highest Abstraction Level- Tasks

These tasks directly lead to the realization of the goal, they are considerably complex and use various combination of skills for their accomplishment.

- <u>Maze Solving Algorithm</u>: This is the highest level of logic, decision algorithms which enable the autonomous system to negotiate the maze in the shortest time possible. Critical decisions like recognizing dead-ends and intersections and the optimum turning direction fall in this category. This involved various skill selection at different stages of the maze.
- <u>Track Position of Robot</u>: The robot should be able to track the time variance of its position with respect to a Global Coordinate System (planer). This will be beneficial in avoiding movement in closed loop trajectories.

2.2 Middle Abstraction Level Skills

Skills are localized decision making event which enable realization of Tasks, they are of a lower complexity level and often involve computation in a specific situation.

- Avoid Collision with walls: The robot is required to maintain a safe distance from the walls at all points of time. This skill will typically be incorporated at all points of the maze and corridor challenge.
- Recognizing Intersections: The robot should be able to detect intersection of walls. In case of a simple turn in the maze, the robot will be able to decide the correct turning direction. A three way intersection will have to be judged based on the position trajectory of the robot.
- Recognizing Dead-Ends: The robot identifies a dead-end and beeps for a door to be opened. It has to wait a certain amount of time and monitor the walls for them to open.
- Detect End of the Maze: The robot should be able to detect the end of the maze once it appears and complete it *eventually*.

2.3 Lowest Abstraction Level Functions

These are the very basic functions of the robot which enable the realization of the skills and eventually the tasks. These are the fundamental abilities of the robot which do not involve development of any algorithms on our part.

- Mobility: The robot is equipped with holonomic wheels which enable it to translate and rotate on a planar field. Due to the nature of the challenge, these actuators are sufficient for the robot to complete the maze.
- Sensors: The robot is equipped with Laser Range Finder and Wide Angle Camera to perceive its environment and be able to relay this information to higher levels of functional abstractions.

3 Components and Specifications

3.1 Hardware Components

3.1.1 Sensors

- Laser Range Finder: A laser range finder uses a laser beam to determine the distance to an object. Here, the laser range finder will be used to calculate the distance between Pico and the walls.
- Wheel encoders/Rotary encoder/Shaft encoder: It is an electro-mechanical device that converts the angular position or motion of a shaft or axle to an analog or digital code. Here, it can be used to get information of robot such as speed, distance and position.

- Wide Angle Camera: With good image processing algorithm, it can be used to localize the position of the robot with respect to the surrounding environment, also to differentiate between walls and door and to interpret different signs (if there is a visual difference between the two).

3.1.2 Computer with intel i7 processor

3.1.3 Actuators

- Wheels & Motors: Omni wheels. The effect is that the wheel can be driven with full force, but will also slide laterally with great ease.
- Pan/ Tilt Head: The head can be moved to look around for obstacles.

4 Software Components

- Ubuntu 14.04: Unix based OS
- Qt Creator: Used to edit and compile C++ codes.

5 Interface

The control of robots can be done by calling the functions in the emc library, which will communicate with ROS topics for data exchange. By running C++ programs, data can be obtained, processed and transmitted from and to certain topics.

- The libraries are: $< emc/rate.h >$ $< emc/io.h >$
- The sensor data can be obtained by: io.readLaserData(), io.readOdometryData().
- The structures of data are saved in: emc::LaserData, emc::OdometryData.
- For driving the robot: io.sendBaseReference & io.sendBaseVelocity.

Figure 5.0.1 shows the interface:

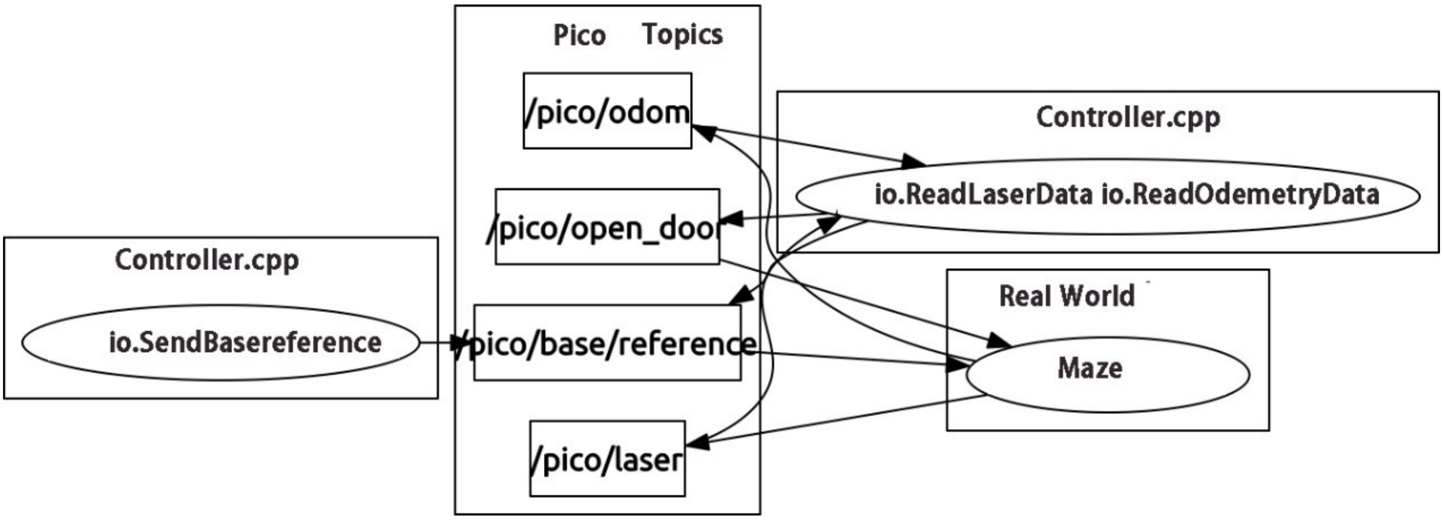

Figure 5.0.1: Interface