Maze Challenge

Group member
Bo Cong
Mian Wei
Petrus Teguh Handoko
Bo Deng
Jian Wen Kok
Zhihao Wu

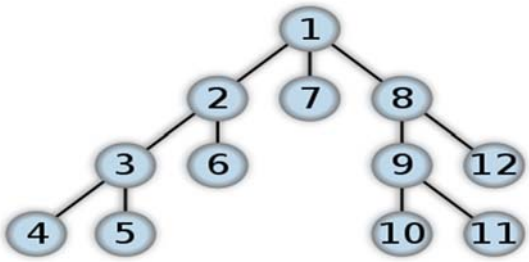
Requirements

- PICO drives autonomously through maze
- PICO should find the exit and the whole robot is across the finish line within 5 minutes.
- PICO is able to deal with approximately axis-aligned walls, open spaces and loops in the maze.
- The task has to be finished within 2 attempts in 7 minutes.
- PICO should not stand still for 30 seconds which counts as an attempt
- PICO may not touch the wall
- The whole PICO should stop within 1.3m to a dead end, and detect whether the dead end is a door.
- PICO should detect every dead it meet
- At the exit PICO should drive forward for 40 cm
- THe software is easy to set-up

Functions (function, working)

Right hand rule and DFS

Depth-First-Search(DFS) is an algorithm for traversing or searching tree or graph data structures. One starts at the root (selecting some arbitrary node as the root in the case of a graph) and explores as far as possible along each branch before backtracking.



Order in which the nodes are visited

[https://en.wikipedia.org/wiki/Depth-first_search]

PICO use Laser Ranger Finder(LRF) detects the maze and build a maze map.

PICO should maintain large velocity (within 0.5m/s) when it is in the middle of the corridor, once it approaches the wall, the velocity decrease

PICO follow the right-hand rule to detect the maze, PICO should drive forward unless it detects a junction, once PICO meet a junction, if PICO has not pass this way, it should turn approximate 90 degree, and drive forward, if PICO has passed this way, then drive forward.

PICO should find all the dead ends of the maze, every time PICO find a dead end and check and ask to open the door and wait for 5 seconds then judge whether there is the right door or not, if not PICO should back to the last cross and go to another way and recursion.

Functions:	
Basic	Actuation： Provide action to the robot's wheel to perform robot movement.
	Detect range： Measure range from the robot using laser sensor.
	Record distance： Measure how long the robot has been travelled using odometer.
Advance skill	Junction check： Determine whether there a junction is present or not.
	Obstacle avoidance： Determine whether there is a wall in the way.
	Door check： Determine whether there is a door.
	Mapping： Record all of the move that the robot has been through.
	Loop Avoidance： Determine whether the robot is going inside the loop.
Main function	Main function: Incorporate all of the skills to achieve the goal.

Components (details, what makes up our design?)

drive control

- Holonomic base (omni-wheels)
- Pan-tilt unit for head

detection

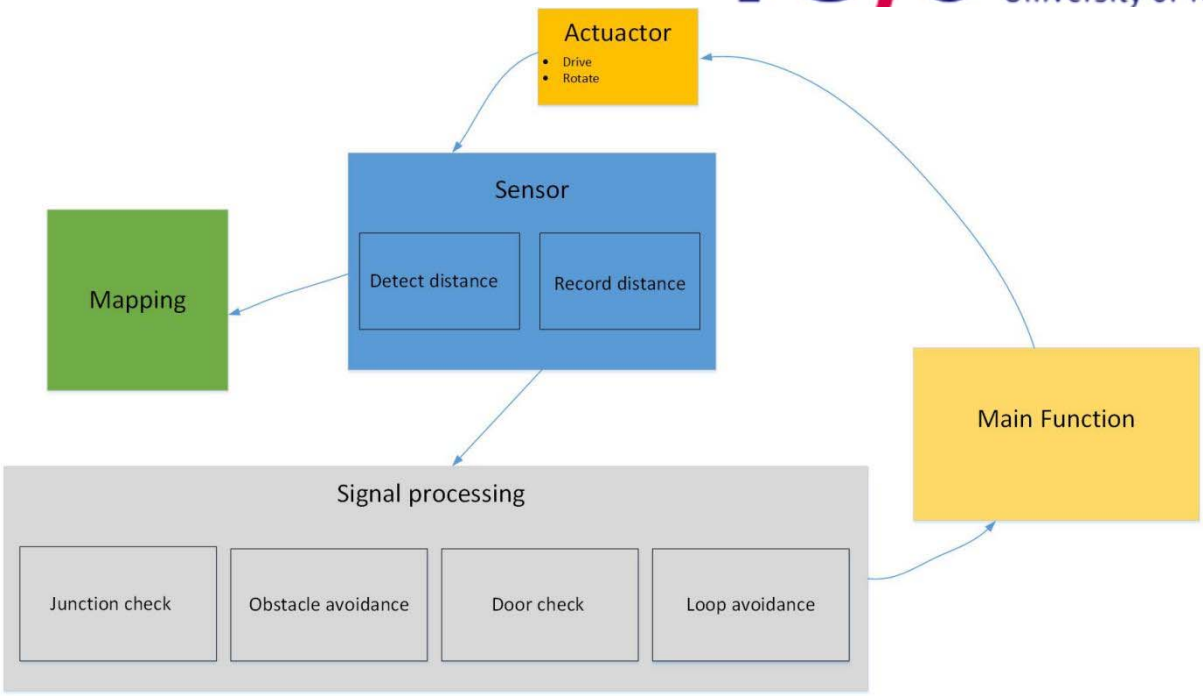
- 170◦ wide-angle camer
- Laser Range Finder (LRF)
- Wheel encoders (odometry)
- Asus Xtion Depth sensor

world model

computer

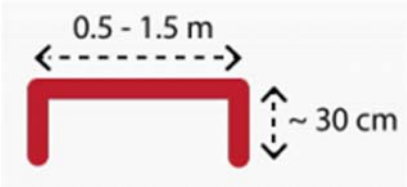
- Intel I7
- Ubuntu 14.04

Schematic of Program Structure



Specifications

- maximum translational speed of 0.5 m/s
- maximum rotational speed of 1.2 rad/s
- Door template: length of 0.5 - 1.5m and with side walls of approximately 30cm, see figure below
- LRF accuracy and range unknown
- odometer accuracy unknown



door template

Interfaces

The odometer and LRF generates data for mapping the environment.

The algorithm sets nodes on the junction as a setpoint for navigation, plans the route and put the actuators to work accordingly.

The odometer and LRF keeps on keeping track of the environment and the software recognizes obstructions, dead ends that might be doors and junction.