EE493 Conceptual Design Report – Taha DOĞAN

**Problem Statement**

In this project, carrying a long object through an open-top maze with two robots, which are not allowed to communicate each other directly, is aimed. The field that robots are carrying a long object is a maze. The maze specifications are given below.

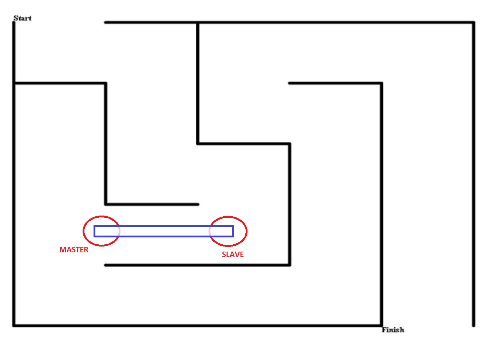
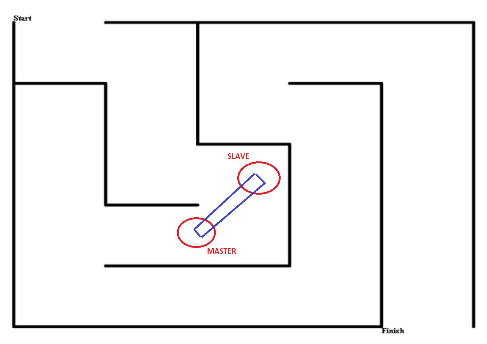
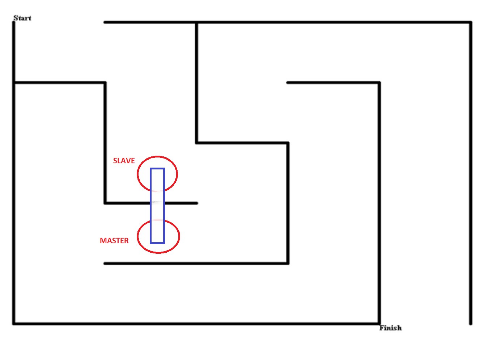
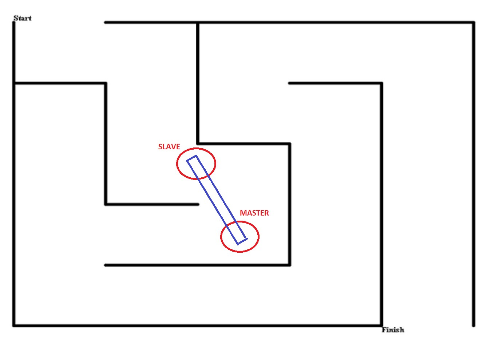
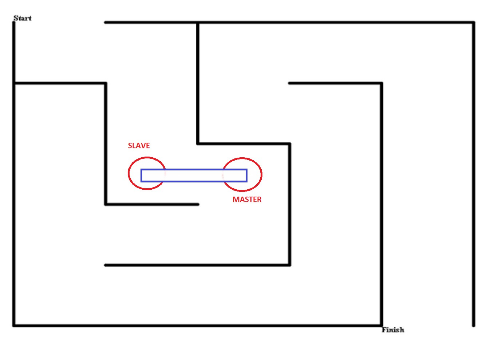
* Maze streets are 250 mm wide
* There are , one entrance and one exit in presence
* Maze pattern is arbitrary However, there is no dead-end.
* Height of the walls: 100 mm. From the bottom of the walls, between 90 mm and 100 mm and the top of the walls are painted black, the rest is white.
* Width of the walls is 10 mm.

When solving the maze “follow the right wall” rule is chosen during the Standard Committee meetings. Although the maze definition is not a real maze due to no dead-end case,

Two robots are expected to carry a long object called *plank*. No sensors or no devices that is mounted directly to the plank is not allowed. Thus, determining and processing the collected data are handled with the devices and sensors that are on the robot itself. The specifications for plank is given below.

* The length of the plank is 500 mm
* Distance between two holders is 400 mm.
* Width of the plank is 50 mm
* Color of the plank is red.
* Plank is elevated 170 mm from the top of the maze platform to the bottom of the plank

According to plank and wall specifications, these two robots cannot handle the *U-turn* at once. In order to turn *U-corners*, robots will perform indirect communication. Thus, the main problem of this project is that observing and determining the other robot’s next move so that two robot can collaboratively proceed in the maze. The illustration of *U-turn* is shown in Figure XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.



(I)

(II)

(III)

(IV)

(V)

Figure XXXXXXXXX: Illustration of handling of *U-turn*

Robots are not allowed to communicate each other directly. Thus, during the maze solving operation, our robot X-Cali performs *sensing*. Both active and passive sensing approaches are applied on X-Cali.

One of the robots are dedicated as master and the other one is slave. Slave observes master so that they can handle such turns.

According to aforementioned physical constraint, the X-Cali maze solver robot proceeds on the path. The interpretation and handling of the path are explained under the Solutions section, in detail.

**Solution I: Image Processing for Collaboration**

In this project, robots should take two roles

1. Master or
2. Slave.

The role of the robots is assigned before they start to carry the plank. A master/slave identification switch does.

**Master Case**

When the robot is in Master role, it leads the way. When the robot faces with a wall, that is a turn, it stays 10 seconds at the same position while adjusting its direction. 10 seconds is chosen in the Standard Committee meetings. In this role, the only thing that the robot is going to do is after a U-turn, it should lead the way till they can arrange the correct angle of the plank it should go back. In Figure XXXXXXXXXXXXXXXXXXXXXXXXXXXXX this event can be observed.

**Slave Case**

Slave Case is the main hardship of this project. Slave should observe and sense the next motion. For this problem, image processing methods are used. The algorithm and procedure are provided below in detail.

1. **Movement**

Movement is the first milestone of the project. Although it seems to be easy to implement, its sustainability is hard to achieve.

In X-Cali, RaspberryPi3 and L293D is used. Motors are two DC Motors. DC motors are chosen, because it provides two options for speed control such as

* PWM
* Voltage Control

In order to drive motors, power supply is used during early stages of the project. After mobilization of the robot is done, a Li-Po battery is going to take place of the power supply.

Motors are not connected directly to the RaspberryPi3. Sinking current for motors from RaspberryPi3 is dangerous for the microprocessor. Thus, by buffering the motors from the control units, we provided safety for the overall circuitry.

X-Cali chosen as rear-wheel drive. The main reason of this can be understood by thinking forklifts. Rear-wheel driving provides more unstability however, this is something we desire. It provides more maneuverability. When maze dimensions are considered, small turn radii have to be achieved. Additionally rear-wheel driving, we use tank-like wheel movements. This phenomena is illustrated in the Figure YYYYYYYYYYYYYYYYY.

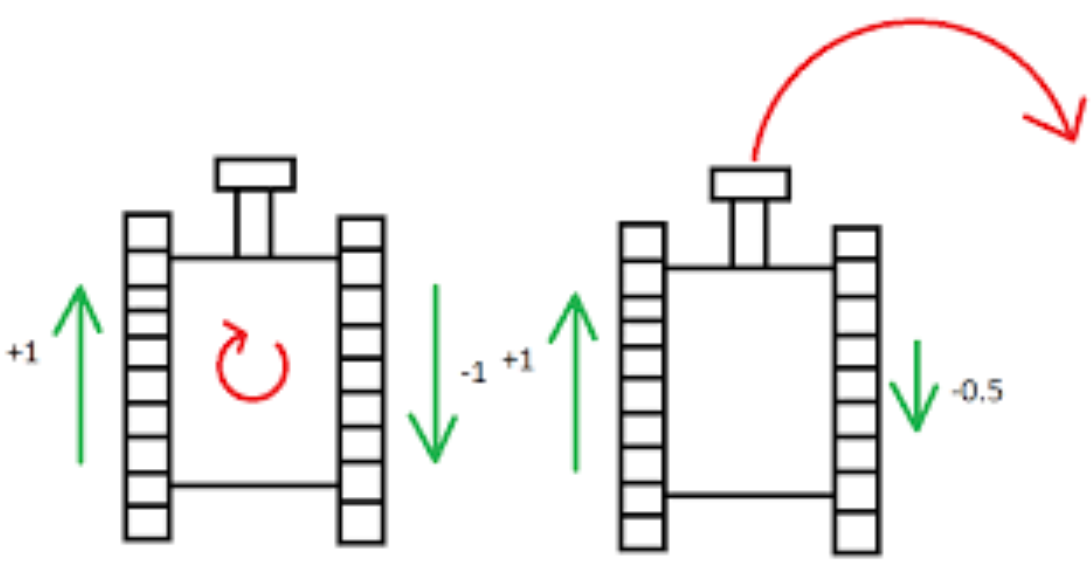


Figure YYYYYYYYYYYYY

A mad-wheel at the front is just used for keeping the balance of the robot.

1. **Sensors**

As sensors, after debating group members, we came up with three options.

1. Ultrasonic Proximity Sensor
2. IR Proximity Sensor
3. Camera

The first two is active sensors. That is ultrasonic proximity sensor sends sound waves and receives the reflected waves. By calculating the difference between emitting time and receiving time, Python program that is embedded in RaspberryPi3 calculates the distance. IR Proximity works with the same principle but rather than sound waves, it uses light.

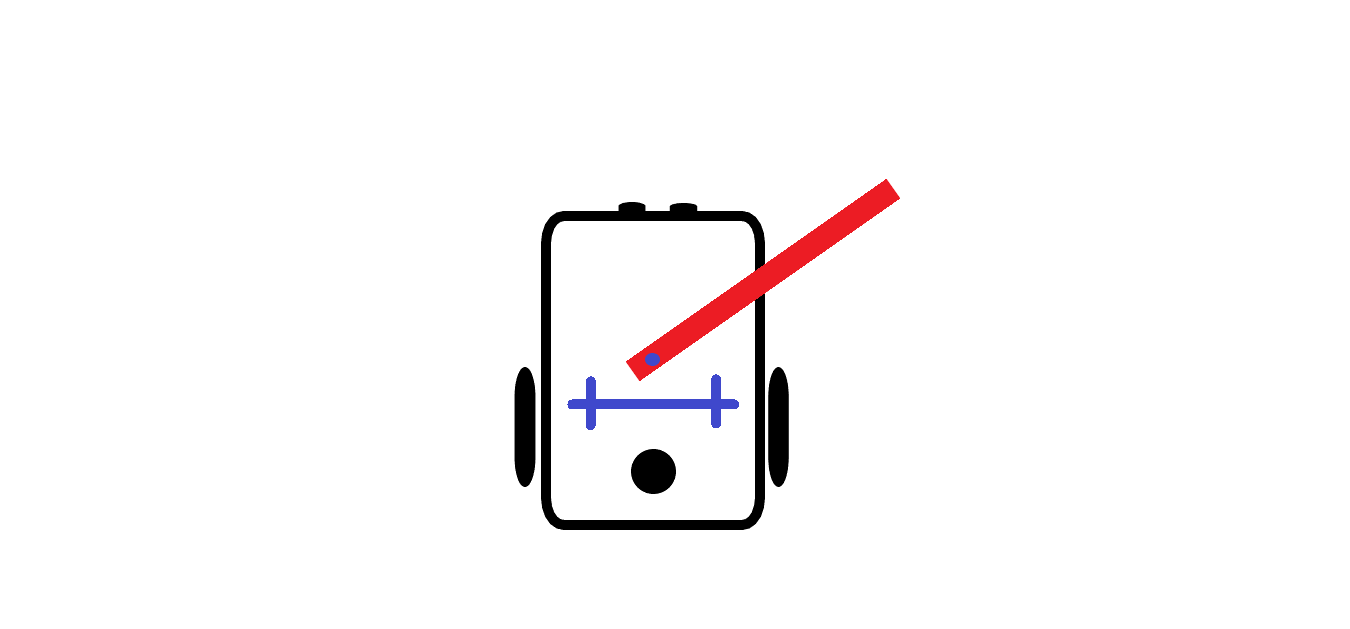
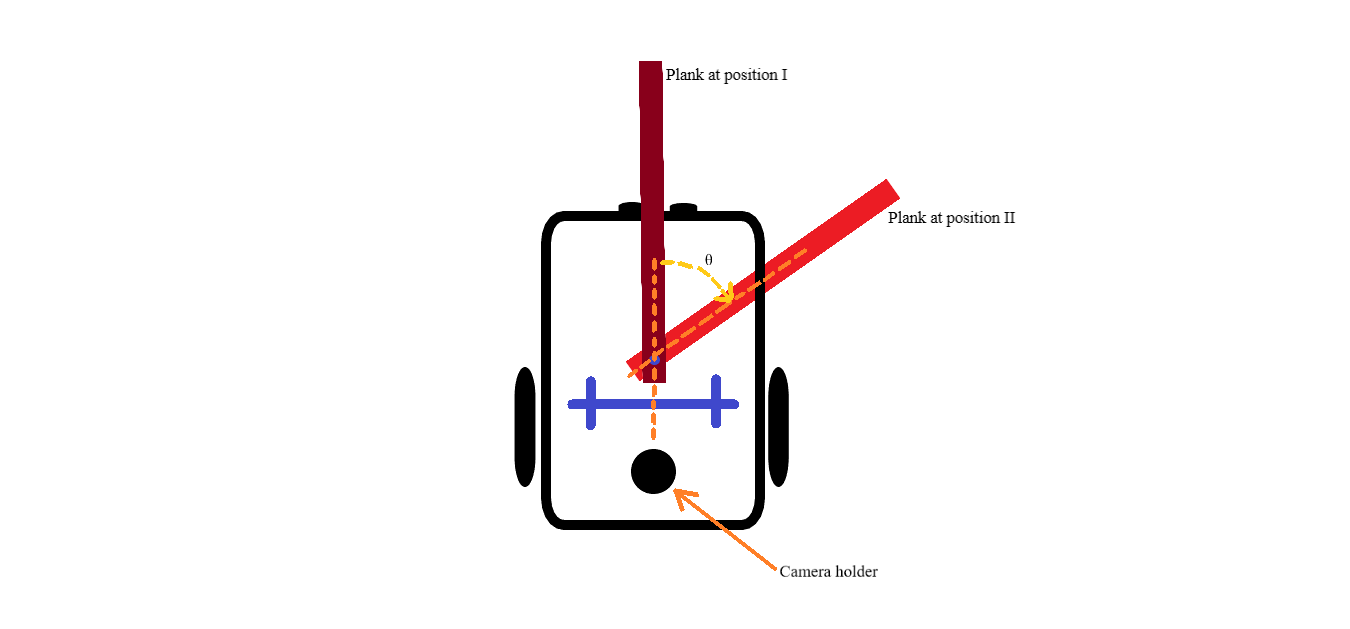
They have both advantages and drawbacks. Sonar sensors measure the distant obstacle with high accuracy, i.e. further than 500 mm. Whereas, IR sensor can measure the close objects, i.e. closer than 100 mm.

The camera is passive sensor, namely it does not change the environment. The first two proximity sensors are used to proceed in maze and to interpret the maze walls. The main sensor that determines the next movement is the camera. The camera is the crucial part of this solution. The algorithm is described under Decision and Image Processing title.

1. **Decision and Image Processing**

As a solution, we came up with use of image processing methods. When the possible movements are considered, the only detection can be done by plank and the other robot’s position. From Standard Committee, color of plank is chosen as red. The top of the walls are white while the rest of the walls are white. Thus, the walls and the plank can be observed easily.

In this solution we are planning to sense the other robot’s movements by measuring the angle of the plank using image processing methods. This is method is shown in Figure ZZZZZZZZZZZZZZZZZZZ.



(1)

(2)

Figure ZZZZZZZZZZZZZZZZZZ

The algorithm is as follows.

1. The frame is slowed down in order not to busy the processor. Probably it will be lowered to 10 frames per second.
2. Using the blue double crosses seen in Figure ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ calibrate the zero angle.
3. Extract the region that we are interested: the plank itself and the joint point.
4. Using Otsu Thresholding Method, binary-mask the image.
5. Find the plank’s direction using tracing the boundaries.
6. Find the line that corresponds to planks orientation.
7. Find the point of intersection of plank’s direction line and the blue calibration line.
8. Using the dot product, find the angle of intersection.
9. If the angle is larger than a previously determined angle value, then this means the other robot is in turn.

As can be seen above, this method lean on a simple idea. This method can be improved by adding the integrating the wall detection algorithms so that robot can observe the other one directly.