Checkpoint 3 Report

[ICN5406] Mobile Robot 2018

Student ID: 0745011 **Name: Petr Gondek** **Date: 21/11/2018**

1. Purpose:

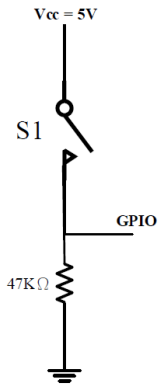
The purpose of this checkpoint is to make sure we can control our robot to move in the arena. The mobile robot needs to detect an obstacle in front of it and take an action to avoid the obstacle in order to continue its motion. Finally, our robot can find the assigned target. In this checkpoint, the target is a ring of LED lights.

2. Description of Design:

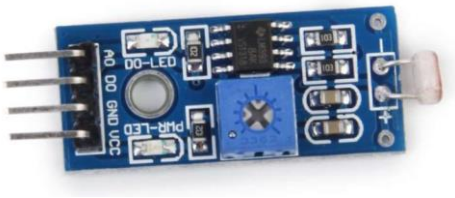
2.1. The Case and HW setup:

First, we created a bumper in the front. We inspired by TA's robot. Under the bumper we hided 2 touch sensors. One at left and one at right. Then we installed the collector with another touch sensor. This collector is catching the puck. We made a hole to the collector and we put there the LED sensor. So, the sensor is on ground level where the light is.

Then we attached the cables to sensors. One to NO so the sensors sends signal when is pressed and the second to 5V input. These two cables we attached to the blue hub. Then we attached them to RPI. One cable went through resistor to ground and second one to GPIO.

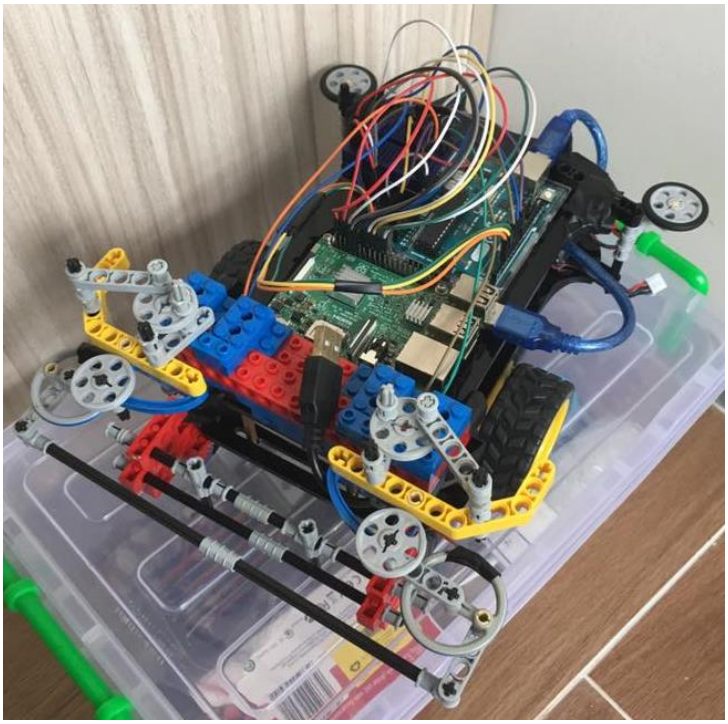


The light sensor was connected: VCC to 5V input, GND to GND at RPI and D0 to GPIO at RPI. This connection was not made direct but also through the blue hub.



At RPI we had pins connected like this:

- Pin 2, 5V input to all touch sensor's ACC
- Pin 4, 5V input to light sensor's ACC
- Pin 7, GPIO to light sensor's D0
- Pin 9, GND to light sensor's GND
- Pins 6, 14 and 20, GNDs for touch sensor's GNDs
- Pin 12, GPIO to left touch sensor's NO
- Pin 16, GPIO to right touch sensor's NO
- Pin 18, GPIO to puck touch sensor's NO



We also added rear wheels to help us to avoid obstacles and walls. It also provides protection for the blue cable between RPI and Arduino and for their connectors.

2.2. The Program

We made three asynchronous running managers: MotorManager, SensorsManager and TimeCallbackManager. These are the lowest layer controllers. MotorManagers sends messages to Arduino to control the movement. The SensorManager uses LightSensor

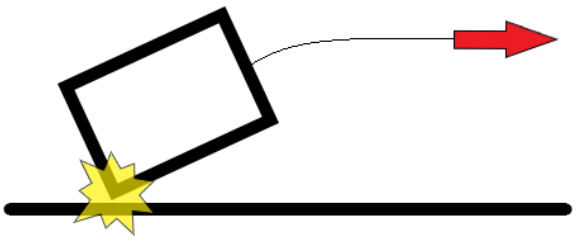
and TouchSensor inherited classes from Sensor. These Sensor classes know how to read the sensor's signal. If any signal from sensors changes its state, the SensorManager sends event to its observers. This is called Observer pattern.

There is another class named Program. The Program class holds the robot AI logic. The class subscribes itself to managers to be sure that it is called when sensor signal changes its state. When an event happens the one of Managers thread calls a Program class method and sets a flag about what happened. We had to use the Condition variables and mutexes to make program run smoothly.

The Main function calls the run() the Program class method in cycle. The run() reads flags and reacts on it. We created own SIGTERM function which stops the robot when ctrl+c is pressed. But suddenly we sometimes end in deadlock.

The robot AI is simple. The robot just goes forward and after while it stops and looks around. It makes more than 360° rotation so next move forward is in different direction. The robot searches the play area randomly. We can easily change our Program class to another. This is feature of State Pattern which we used.

When robot hits the obstacle, it recognizes which sensor is pressed. If left, right or both. If it is only left or right, the robot moves backward and turns a little bit right or left. It can be seen from the image below. If both sensors are triggered the robot moves straight backward.



3. Result

The robot made third checkpoint contest very well. The solution to go little bit forward and look around was good concept for this checkpoint. Also, the solution to recognize which touch sensor was triggered and do a curve backward move was very smart. It helped a lot.

4. Discussion

We must program better starting moves. We can predict the starting position and go more forward. If we hit the obstacle, we can go around it and then we can assume that

we are almost in the middle of arena. At this point we can look around for the puck. This will be better solution for last contest