Checkpoint # 3 Report

1. Purpose:

The primary goal of this checkpoint is to meticulously configure the sensors on the mobile robot's base. This setup empowers the robot with the ability to detect collisions with obstacles and perceive ambient light through touch sensors and light sensors. Following this configuration, the robot is poised to accomplish a straightforward mission—locating a light ball. The mission involves initiating contact with an obstacle, subsequently relying on the sensors to identify the presence of light and ultimately pinpointing the target object.

2. Description of Design:

For the sake of convenience in testing and debugging, this checkpoint focuses solely on implementing the Arduino board. It is important to note that this work does not involve any communication with the Raspberry Pi using ROS. The Arduino code is specifically designed to detect collisions with touch sensors, directly connecting to Arduino pins with analog input. The same approach is taken for the light sensor.

Upon collision detection, the robot initiates a rotation for a predetermined duration, approximately 5 seconds. During this rotation, if the robot identifies a light value surpassing a predefined threshold, it pauses and assumes control to capture the target. In the absence of a detected target during the rotation time, the robot pauses momentarily and proceeds to move straight ahead until it encounters a new obstacle. The process then recommences, repeating the aforementioned sequence.

The touch sensor located at the base's bottom plays a crucial role in mission completion. Once this touch sensor makes contact with the ball, signaling the successful completion of the mission, the robot comes to a halt. The code implementation details are outlined below:

Arduino:

(1) Checkpint3.ino

```
// include
#include<ros.h>
#include<std_msgs/Int32.h>

// pins
#define EnA 5
#define EnB 6
#define In1 8
#define In2 9
#define In3 10
#define In4 11
#define In5 7  // ball
#define In6 12 // left
#define In7 13 // right
#define Light A5 // light
```

```
// ros
ros::NodeHandle nh;
std_msgs::Int32 light;
ros::Publisher publisher("light_feedback", &light);

// flags and time
int finding = 0;
int right_collision,left_collision = 0;
int finding_ball = 0;
int find_ball = 0;
int find_ball_sec = 0;
int catch_ball = 0;

void move_robot(int right,int left)
{
    if(right>0)
    {
        digitalWrite(In1, LOW);
        digitalWrite(In2, HIGH);
    }
    else
    {
        digitalWrite(In1, HIGH);
        digitalWrite(In2, LOW);
        right *= -1;
    }
    if(left>0)
    {
        digitalWrite(In3, LOW);
        digitalWrite(In4, HIGH);
    }
    else
    {
        digitalWrite(In3, HIGH);
        digitalWrite(In4, LOW);
        left *= -1;
    }
    analogWrite(EnA, right);
    analogWrite(EnB, left);
}

void foward()
{
    move_robot(105,100);
}

void backward()
{
```

```
    move_robot(-105,-100);
}

void right()
{
    move_robot(100,135);
}

void left()
{
    move_robot(135,100);
}

void left_spin()
{
    move_robot(100,0);
}

void right_spin()
{
    move_robot(0,100);
}

void robot_stop()
{
    move_robot(0,0);
}

void collision()
{
    int sec = 0;
    while(sec <= 10)
    {
        backward();
        sec++;
        delay(100);
    }
}

void robot_pause()
{
    int sec = 0;
    while(sec <= 10)
    {
        robot_stop();
        sec++;
        delay(100);
    }
```

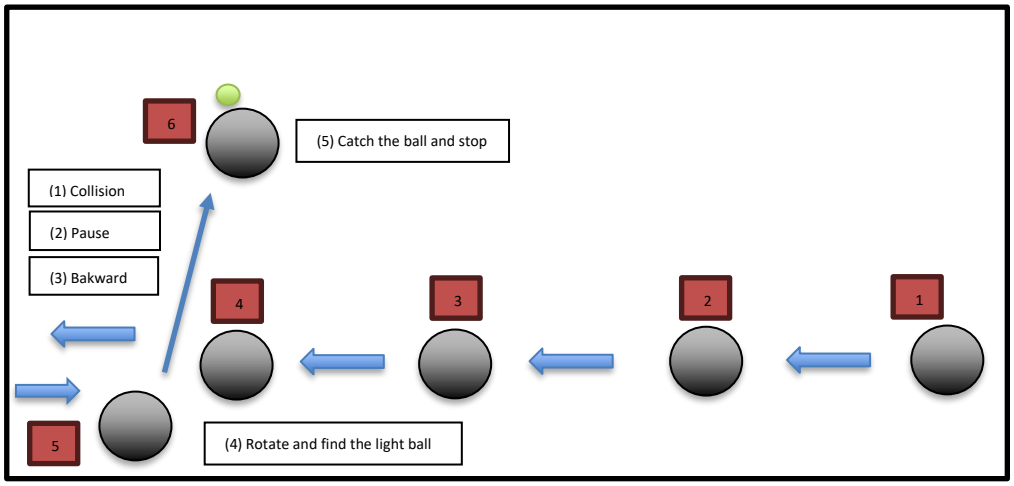
```
}

void setup()
{
    //ros
    nh.initNode();
    nh.advertise(publisher);
    Serial.begin(57600);

    //pin
    pinMode(EnA, OUTPUT);
    pinMode(EnB, OUTPUT);
    pinMode(In1, OUTPUT);
    pinMode(In2, OUTPUT);
    pinMode(In3, OUTPUT);
    pinMode(In4, OUTPUT);
    pinMode(In5, INPUT);
    pinMode(In6, INPUT);
    pinMode(In7, INPUT);
    pinMode(Light ,INPUT);

    //flags
    finding = 1;
    right_collision = 0;
    left_collision = 0;
    finding_ball = 0;
    find_ball = 0;
    find_ball_sec = 0;
    catch_ball = 0;
}

void loop()
{
    int ball_press = digitalRead(In5);
    int left_press = digitalRead(In6);
    int right_press = digitalRead(In7);
    int light_data = analogRead(Light);

    //ball touch sensors
    if (ball_press == LOW)
    {
        catch_ball = 1;
    }

    //left touch sensor
    if(left_press == LOW)
    {
        left_collision = 1;
```

```
    }

    //right touch sensor
    if(right_press == LOW)
    {
        right_collision = 1;
    }

    //light sensor
    if(light_data < 250)
    {
        find_ball = 1;
    }

    //counting finding ball time
    if(finding_ball == 1)
    {
        find_ball_sec++;
        if(find_ball_sec == 100)
        {
            find_ball = 1;
            find_ball_sec = 0;
        }
    }

    //whole finding process
    if(catch_ball == 1)
    {
        robot_stop();
    }
    else
    {
        if(finding == 1 && right_collision == 0 && left_collision == 0)
        {
            foward();
        }
        else if(finding == 1 && right_collision == 1 && left_collision
== 0)
        {
            find_ball = 0;
            robot_pause();
            collision();
            robot_pause();
            finding = 0;
            finding_ball = 1;
        }
        else if(finding == 1 && right_collision == 0 && left_collision
== 1)
```

```
        {
            find_ball = 0;
            robot_pause();
            collision();
            robot_pause();
            finding = 0;
            finding_ball = 1;
        }
        else if(finding == 1 && right_collision == 1 && left_collision
== 1)
        {
            find_ball = 0;
            robot_pause();
            collision();
            robot_pause();
            finding = 0;
            finding_ball = 1;
        }
        else if(finding == 0 && right_collision == 1 && left_collision
== 0)
        {
            left_spin();
            if(find_ball == 1)
            {
                robot_pause();
                finding = 1;
                right_collision = 0;
                left_collision = 0;
                finding_ball = 0;
            }
        }
        else if(finding == 0 && right_collision == 0 && left_collision
== 1)
        {
            left_spin();
            if(find_ball == 1)
            {
                robot_pause();
                finding = 1;
                right_collision = 0;
                left_collision = 0;
                finding_ball = 0;
            }
        }
        else if(finding == 0 && right_collision == 1 && left_collision
== 1)
        {
            left_spin();
```

```
            if(find_ball == 1)
            {
                robot_pause();
                finding = 1;
                right_collision = 0;
                left_collision = 0;
                finding_ball = 0;
            }
        }
    }
    delay(100); // unit is ms => delay 0.1sec => 10HZ
}
```

3. Result:



4. Discussion:

The provided Arduino code demonstrates a meticulous implementation for a mobile robot with touch sensors and a light sensor, emphasizing collision detection and light-based navigation. Several noteworthy insights emerge from the code.

(1) Motor Control and Actuation:

The code employs a judicious combination of digitalWrite and analogWrite functions to precisely control the robot's motors. The specific pin configurations and analogWrite values enable the execution of various movements, including forward, backward, and rotational actions. This underscores the significance of robust motor control in achieving desired robot behaviors.

(2) Sensor Integration for Environmental Perception:

The incorporation of touch sensors (left, right, and ball) enables the robot to detect collisions and interactions with its surroundings. Through digital readings from these sensors, the code orchestrates distinct responses to collisions and effectively utilizes the ball touch sensor to signify mission completion. This robust sensor integration is pivotal for enhancing the robot's awareness and responsiveness.

(3) Light Sensing Capabilities:

The inclusion of a light sensor adds a layer of environmental perception to the robot. By reading analog values, the robot can assess ambient light conditions. The

establishment of a predefined threshold value showcases a thoughtful approach to light-based decision-making. Upon detecting a target during rotation, the robot adeptly pauses, exemplifying the importance of environmental awareness in mission execution.

In conclusion, the presented Arduino code manifests a comprehensive approach to autonomous robot control, encapsulating motor actuation, sensor integration, and responsive decision-making. The careful design and utilization of touch and light sensors empower the robot with the capability to navigate its environment, detect obstacles, and successfully accomplish a mission. This comprehensive system ensures effective collision detection, light sensing, and mission completion without the need for ROS communication in this particular checkpoint. This work lays a foundation for further exploration and refinement, particularly when extending the robot's functionalities and integrating additional sensors for enhanced autonomy.