ENGINEERING JOURNAL

COMP20170
Introduction To Robotics

- Ronit – 18204009 Taran - 18203372 Krishna – 18203023 Enamuel - 8729365

26th January 2020

- **Our goals for today:**
  - To think about possible solutions to the actions the robot had to take in each room.
  - To figure out how the robot would now which room it was currently in.
  - To start coding the functions to be used for each room.

- **Our accomplishments today**
  - We managed to make the robot detect the fire area and reverse into it.
  - We got the robot to play a sound as soon as it detected it was in a blank room.
  - We made progress towards making a function that determines the starting position and thus the order in which the robot would enter the rooms.

- **Some difficulties we encountered:**
  - We tried to use gyro sensors to make accurate degree turns but the sensors themselves were not accurate.
  - We tried to use sonar sensors to measure distances from the walls/box/survivor to the robot, but they time to register and often didn't register at all.

27th February 2020

- **Our goals for today**
  - To start implementing the code for the actions the robot was to take:
  - In the "box room"
  - In the "survivor room"
  - To improve our rough versions of the code for the:
  - "Fire room"
  - "Blank room"

- **Our accomplishments today:**
  - Made a rough implementation of the code needed to complete the box and survivor room.
  - After completing the necessary actions in the fire and blank rooms, the robot moved to the next rooms
  - We modularised the code for the fire and blank rooms into one function called emptyOrRedRoom.
  - The robot measures the distance in front and behind it and determines if the border is on its left or right. These allow the robot to determine which room it is currently in.

- **Some difficulties we encountered:**
  - Struggling to make to robot continue from one room to another as it wouldn't move in a straight line.
  - We tried to modularise gyro right turns using the following code, but it wouldn't work:

```
void gyroTurnRight(int degrees)
{
  resetGyro(S2);
  repeatUntil(getGyroDegrees(S2) > degrees)
  {
    setMotorSpeed(motorC, -25);
    setMotorSpeed(motorB, 25);
  }

  setMotorSpeed(motorC,0);
  setMotorSpeed(motorB,0);
  sleep(2000);
}
```

  - It took a lot of trials to get the right values needed for robot movement forwards and backwards in certain scenarios, e.g. moving forward and back to pick up the survivor.
  - Since the sonar sensor took time to register distances, it often didn't register the correct distance/room and performed incorrect procedures for that room

28th February 2020

- **Our goal for today:**
  - To test to robot on the course some more and
  - To review changes that need to be made to improve functionality

- **Our accomplishments today:**
  - We finished the function called startingPosition which found the first room the robot was to enter and therefore the procedure to be carried out for the entire course.

- **Some difficulties we encountered:**
  - We had to replace each of the following function calls with the code from the body of the gyroTurnRight() function above:

```
gyroTurnRight(90);
gyroTurnLeft(-90);
```

4th February 2020

- **Our goals for today:**
  - To implement the final changes to the code and ensure the robot knows which procedure to run in each room.

- **Our accomplishments for today:**
  - We improved each function for each room by adding an if else statement that enables the robot to make the correct turns in each room.

- **Some difficulties we encountered:**
  - We had to try many different integer values for turns, movement and sonar measurements to get the robot to move properly.

****

Personal role(s) or contribution(s) to the group effort:

- **Taran:** developed the code for the startingPosition(), isWallOnRight() and emptyOrRedRoom() functions as well as the main function. Helped with testing the robot and made a board setup at home for further testing.

- **Emanuel:** developed the code for the boxRoom() and survivorRoom() functions. Also helped with testing the robot.

- **Ronit:** Robot code tested to ensure proper functionality and formatted the code into functions. Recorded the robot completing the course at a few different starting positions. Edited the video and the documentation of the journal.

- **Krishna:** Helped with code commenting along with debugging errors in our code. Also helped with testing and solution for the same. Helped with converting code into functions.

****

Function **isWallOnRight**

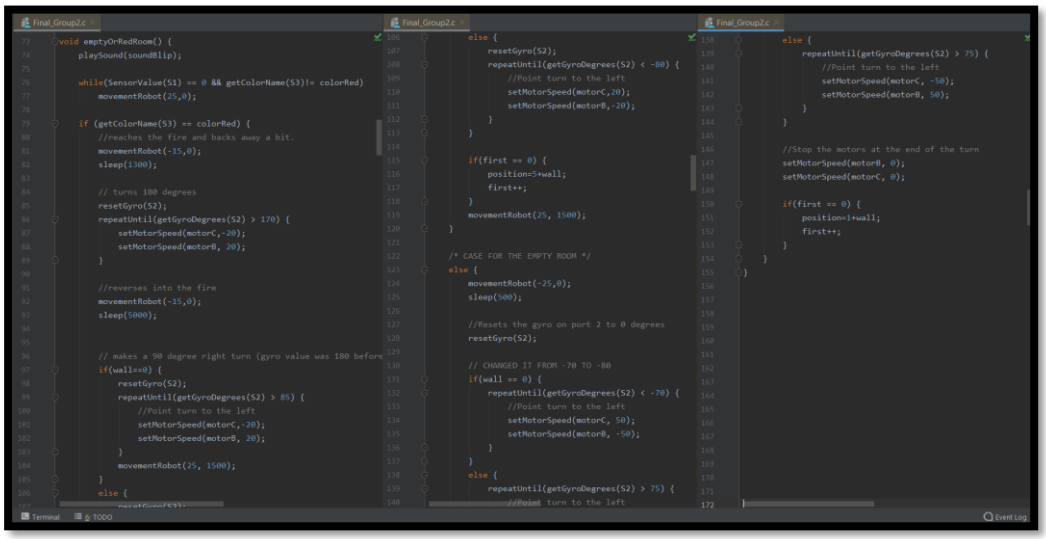
```
/* Helps detect whether the movement should be clockwise or anti-clockwise */
void isWallOnRight() {

    if(SensorValue(S4)<20)
        wall = 0;
    else {
        playSound(soundUpwardTones);
        wall = 1;
    }
}
```

Explanation

During the startingPosition() code when the robot is doing gyro rotations to determine the first room it encounters, it also calls this function and assigns the global variable “wall” the value 0 if the sonar sensor detects an object(in this case a wall) within 20 cm of the robot. 1 is assigned to wall otherwise. The value of wall determines in which direction the robot will turn in each room.

****

Function **emptyOrRedRoom**



```
void emptyOrRedRoom() {
    playSound(soundBlip);

    while(SensorValue(S1) == 0 && getColorName(S3)!= colorRed)
        movementRobot(25,0);

    if (getColorName(S3) == colorRed) {
        //reaches the fire and backs away a bit.
        movementRobot(-15,0);
        sleep(1300);

        // turns 180 degrees
        resetGyro(S2);
        repeatUntil(getGyroDegrees(S2) > 170) {
            setMotorSpeed(motorC,-20);
            setMotorSpeed(motorB, 20);
        }

        //reverses into the fire
        movementRobot(-15,0);
        sleep(5000);


        // makes a 90 degree right turn (gyro value was 180 before
        if(wall==0) {
            resetGyro(S2);
            repeatUntil(getGyroDegrees(S2) > 85) {
                //Point turn to the left
                setMotorSpeed(motorC,-20);
                setMotorSpeed(motorB, 20);
            }
            movementRobot(25, 1500);
        }
        else {
```

```
        else {
            resetGyro(S2);
            repeatUntil(getGyroDegrees(S2) < -80) {
                //Point turn to the left
                setMotorSpeed(motorC,20);
                setMotorSpeed(motorB,-20);
            }
        }

        if(first == 0) {
            position=5+wall;
            first++;
        }
        movementRobot(25, 1500);
    }

    /* CASE FOR THE EMPTY ROOM */
    else {
        movementRobot(-25,0);
        sleep(500);

        //Resets the gyro on port 2 to 0 degrees
        resetGyro(S2);

        // CHANGED IT FROM -70 TO -80
        if(wall == 0) {
            repeatUntil(getGyroDegrees(S2) < -70) {
                //Point turn to the left
                setMotorSpeed(motorC, 50);
                setMotorSpeed(motorB, -50);
            }
        }
        else {
            repeatUntil(getGyroDegrees(S2) > 75) {
                //Point turn to the left
```

```
        else {
            repeatUntil(getGyroDegrees(S2) > 75) {
                //Point turn to the left
                setMotorSpeed(motorC, -50);
                setMotorSpeed(motorB, 50);
            }
        }

        //Stop the motors at the end of the turn
        setMotorSpeed(motorB, 0);
        setMotorSpeed(motorC, 0);

        if(first == 0) {
            position=1+wall;
            first++;
        }
    }
}
```

Explanation

This function enables the robot to carry out the procedures in the red room and the empty room. The robot moves forward and if it encounters the fire first it performs the operations for the red room. If the touch sensor encounters a wall the code for the empty room is run. In the red room as soon as the robot detects the fire, it makes a point turn and reverses into the fire area. It then moves on to the next room. In the empty room, the robot plays a sound and then moves on to the next room.

****

Function **startingPosition**

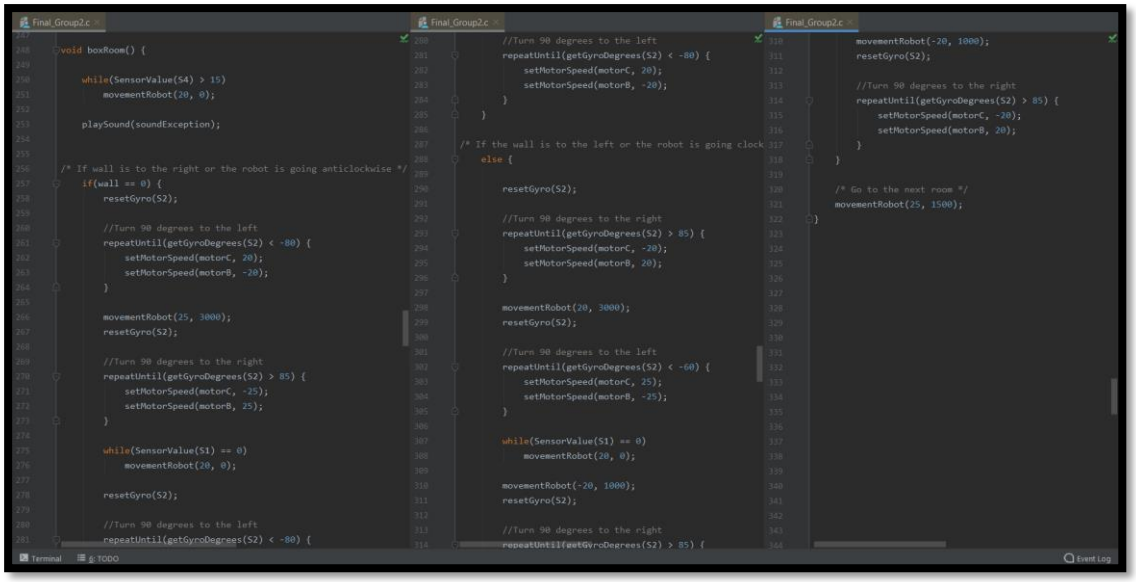
```
/* This code tells us what starting point we are at 1,2,3 or 4 */
void startingPosition() {
    int sum=0;
    int distance1,distance2;

    distance1 = SensorValue(S4);

    resetGyro(S2);
    //turns 180 degrees to check the value for opposite side
    repeatUntil(getGyroDegrees(S2) > 85) {
        setMotorSpeed(motorC,-25);
        setMotorSpeed(motorB, 25);
    }

    setMotorSpeed(motorC,0);
    setMotorSpeed(motorB,0);
    sleep(2000);

    isWallOnRight();

    resetGyro(S2);

    //turns 180 degrees to check the value for opposite side
    repeatUntil(getGyroDegrees(S2) > 85) {
        setMotorSpeed(motorC,-25);
        setMotorSpeed(motorB, 25);
    }
    setMotorSpeed(motorC,0);
    setMotorSpeed(motorB,0);
    sleep(2000);

    distance2= SensorValue(S4);
    sum=distance1+distance2;
    resetGyro(S2);
    repeatUntil(getGyroDegrees(S2) > 180) {
        setMotorSpeed(motorC,-25);
        setMotorSpeed(motorB, 25);
    }
    setMotorSpeed(motorC,0);
    setMotorSpeed(motorB,0);
    sleep(2000);


    //the sum of the distances is different from each point hence helping us distinguish the points we are starting at.


    if(sum>100) {
      if(wall==0)
          emptyOrRedRoom();
      else if(distance1<40)
          position=6;
      else
          position=2;
    }

    else if(sum>40 && sum <60)
        position=3+wall;
    else if(sum>60 && sum<80)
        position=7+wall;
    else
        startingPosition();
}
```

Explanation

The robot identifies the first room before it enters it and at the same time it finds out the order in which each of the rooms appear. Using the sonar sensor, it measures the distance between the closest object in front of it (whether it be a wall, survivor or box). It makes two gyro point turns, one to determine if the border/wall of the board is to the right of the robot and another to measure the distance between itself and the closest object behind it. The two distances are added together and the distance range that sum falls into determines the starting position.

****

Function **boxRoom**



```
void boxRoom() {

    while(SensorValue(S4) > 15)
        movementRobot(20, 0);

    playSound(soundException);


 /* If wall is to the right or the robot is going anticlockwise */
    if(wall == 0) {
        resetGyro(S2);

        //Turn 90 degrees to the left
        repeatUntil(getGyroDegrees(S2) < -80) {
            setMotorSpeed(motorC, 20);
            setMotorSpeed(motorB, -20);
        }

        movementRobot(25, 3000);
        resetGyro(S2);

        //Turn 90 degrees to the right
        repeatUntil(getGyroDegrees(S2) > 85) {
            setMotorSpeed(motorC, -25);
            setMotorSpeed(motorB, 25);
        }

        while(SensorValue(S1) == 0)
            movementRobot(20, 0);

        resetGyro(S2);

        //Turn 90 degrees to the left
        repeatUntil(getGyroDegrees(S2) < -80) {
        //Turn 90 degrees to the left
        repeatUntil(getGyroDegrees(S2) < -80) {
            setMotorSpeed(motorC, 20);
            setMotorSpeed(motorB, -20);
        }
    }
/* If the wall is to the left or the robot is going clock
    else {

        resetGyro(S2);

        //Turn 90 degrees to the right
        repeatUntil(getGyroDegrees(S2) > 85) {
            setMotorSpeed(motorC, -20);
            setMotorSpeed(motorB, 20);
        }

        movementRobot(20, 3000);
        resetGyro(S2);

        //Turn 90 degrees to the left
        repeatUntil(getGyroDegrees(S2) < -60) {
            setMotorSpeed(motorC, 25);
            setMotorSpeed(motorB, -25);
        }

        while(SensorValue(S1) == 0)
            movementRobot(20, 0);

        movementRobot(-20, 1000);
        resetGyro(S2);

        //Turn 90 degrees to the right
        repeatUntil(getGyroDegrees(S2) > 85) {
        movementRobot(-20, 1000);
        resetGyro(S2);

        //Turn 90 degrees to the right
        repeatUntil(getGyroDegrees(S2) > 85) {
            setMotorSpeed(motorC, -20);
            setMotorSpeed(motorB, 20);
        }
    }

    /* Go to the next room */
    movementRobot(25, 1500);
}
```

Explanation

This function is run when the robot enters the box room. It moves forward until it is 10 centimetres from the box. Using the global “wall” variable, it finds out if the border of the board is to the right. Therefore, the robot knows if it needs to turn left first or right first. After avoiding the box, it moves on to the next room.

****

Function **survivorRoom**

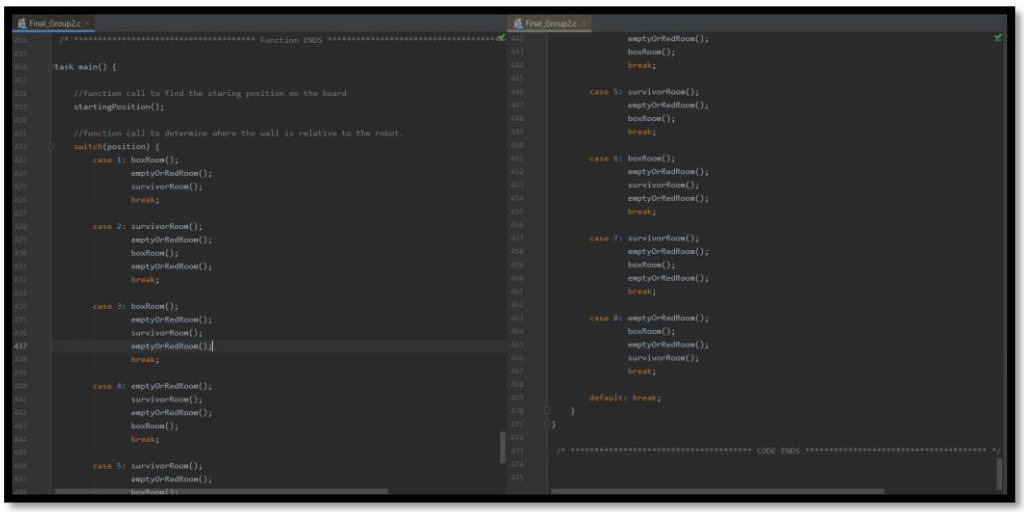
```
void survivorRoom() {

    while(SensorValue(S4) > 10)
        movementRobot(20, 0);

    movementRobot(-13, 570);
    moveMotor(arm, -180, degrees, 40);
    sleep(300);

    movementRobot(20, 1500);
    moveMotor(arm, 180, degrees, 40);

    /* Added the following if else statement */
    /* If wall is to the right */
    if(wall == 0) {
        resetGyro(S2);

        //Full 90 to the left
        repeatUntil(getGyroDegrees(S2) <-60) {
            setMotorSpeed(motorC, 20);
            setMotorSpeed(motorB, -20);
        }

        setMotorSpeed(motorC, 0);
        setMotorSpeed(motorB, 0);
        sleep(2000);

        //Go to the next room
        movementRobot(30, 4000);
    }

    else {
        resetGyro(S2);
        setMotorSpeed(motorB, 0);
        sleep(2000);

        //Go to the next room
        movementRobot(30, 4000);
    }

    else {
        resetGyro(S2);

        //Turn 90 degrees to the right
        repeatUntil(getGyroDegrees(S2) > 85) {
            setMotorSpeed(motorC, -20);
            setMotorSpeed(motorB, 20);
        }

        setMotorSpeed(motorC, 0);
        setMotorSpeed(motorB, 0);
        sleep(2000);

        //Move to the next room
        movementRobot(30, 2000);
    }
}
```

Explanation

This function runs when the robot enters the survivor room. It moves forward until it is 10 centimetres away from the survivor. It moves back a bit, lowers its arm, moves forward a bit and picks up the survivor. Then it turns left or right depending on if the border of the board is to the left or right of the robot itself. After that it moves forward to the next room.

****

Function **MAIN**



```
Final_Group2.c
/* ********************************** Function ENDS ********************************
task main() {

    //function call to find the staring position on the board
    startingPosition();

    //function call to determine where the wall is relative to the robot.
    switch(position) {
        case 1: boxRoom();
                emptyOrRedRoom();
                survivorRoom();
                break;

        case 2: survivorRoom();
                emptyOrRedRoom();
                boxRoom();
                emptyOrRedRoom();
                break;

        case 3: boxRoom();
                emptyOrRedRoom();
                survivorRoom();
                emptyOrRedRoom();
                break;

        case 4: emptyOrRedRoom();
                survivorRoom();
                emptyOrRedRoom();
                boxRoom();
                break;

        case 5: survivorRoom();
                emptyOrRedRoom();
                boxRoom();
```

```
Final_Group2.c
                emptyOrRedRoom();
                boxRoom();
                break;

        case 5: survivorRoom();
                emptyOrRedRoom();
                boxRoom();
                break;

        case 6: boxRoom();
                emptyOrRedRoom();
                survivorRoom();
                emptyOrRedRoom();
                break;

        case 7: survivorRoom();
                emptyOrRedRoom();
                boxRoom();
                emptyOrRedRoom();
                break;

        case 8: emptyOrRedRoom();
                boxRoom();
                emptyOrRedRoom();
                survivorRoom();
                break;

        default: break;
    }
}

/* ********************************** CODE ENDS ********************************** */
```

In task main, the startingPosition() function is called so the robot knows which room it will enter first. Since the global variable “position” was assigned a value in the startingPosition() function it gets passed to a switch statement. “Position” could have been assigned values 1 to 8 each corresponding to a different course and the chosen course is the one that the robot performs for the run.

****