

CZ3004-Multi-Disciplinary Project

Log Report 2 (Implementation)

Submitted by

Team 2

|  |  |
| --- | --- |
| Member 1: Aye Pwint Phyu | Matric No. : U1920903F |
| Member 2: Lin Yan | Matric No. : U1920925F |
| Member 3: Jesline Ng | Matric No. : U1920213C |
| Member 4: Phone Myint Thu Mya Min | Matric No. : U1920225E |

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

NANYANG TECHNOLOGICAL UNIVERSITY

|  |  |  |
| --- | --- | --- |
|  | Member | Section |
| 1 | Aye Pwint Phyu | Robot movement and Rpi communication |
| 2 | Lin Yan | Image recognition |
| 3 | Jesline Ng | Android development |
| 4 | Phone Myint Thu Mya Min | Algorithm |

# 1) Robot movement and Rpi communication

**Implementation of the design proposed earlier**

Basic movements such as Forward, Reverse, Turn Right and Turn Left are implemented using timed loops. Specific data packets are written into the serial port accordingly. An example of a data packet with speed 130 forward movement is [0x5a, 0x0c, 0x01, 0x01, ***0x00****,* ***0x82***, 0x00, 0x00, ***0x00****,* ***0x00***, 0x00, 0xff]. The first bolded pair of Hex values are considered as 16-bit data for speed and the second as turning angle. For the speed, positive values indicate forward, and negative values indicate reverse. For the turning angle, positive values indicate left and negative values indicate right. The functions will then be called into counted loops to move about depending on the nature of the call. If faster speeds are required, the time set for the timed loop can be reduced to match the same move distance/angle.

For easier understanding, written below is the sudo code explanation for such movements:

def forward/reverse10cm ()

move = [specific speed with zero turning angle]

start = start time

while True:

write move into the port using serial communication

if (current time - start) >= certain value (in seconds): break

def turning90deg ()

turn = [positive speed and max turning angle]

reverse = [negative speed and max turning angle]

turnv2 = [positive speed and very low turning angle]

start = start time

# first turn with max turning angle

while True:

write turn into the port using serial communication

if (current time - start) >= certain value (in seconds): break

# reverse with same turning angle

while True:

write reverse into the port using serial communication

if (current time - start) >= certain value (in seconds): break

# turn with reduce angle to adjust the full 90 degree turn

while True:

write turnv2 into the port using serial communication

if (current time - start) >= certain value (in seconds): break

For Rpi communication, there are software communication via remote wifi access and Bluetooth to communicate with other devices and hardware communication with the camera.

* Remote wifi access was established after setting the static ip address as 192.168.2.1 as our group is group (2). Other devices can connect to Rpi’s wifi or remote access via VNC or SSH directly. Here are the details needed for other devices to know for successful access.
  + Wifi name - MDPGrp2
  + Wifi password - 2021Grp2
  + Login username - pi
  + Login user password - group2
* Bluetooth connection can be established by running a Bluetooth program. Upon a successful connection, Rpi and other devices can exchange messages to and fro. In this Bluetooth program,
  + Python’s standard socket module is utilized to establish both server and client in one program.
  + Incoming messages are decoded into ‘utf-8’ format to display/store as string messages.
  + Outgoing messages are sent as string messages as well.
  + The reliability of these message exchanges is going to be crucial for the next step of the project: *Integration*.
* Camera connection was achieved by temporarily attaching it to the robot to test the OpenCV program written by my teammate: Lin Yan. After successful connection and multiple testings with the robot for successful detection of the images provided, the camera was then detached again for Lin Yan to improve her program further for the next phase of the project: *Integration*.

**Unexpected issues and changes incorporated**

For Robot movement, as we are testing the robot in different places with different floor textures, the inaccuracy of the displacements is encountered due to the different friction level. As of this phase, we are unable to fix the issue. However, in the next phase with integration, we will be able to use camera and adjust the offset of the displacements by differentiating the size of the pictures taken (i.e., bigger picture indicates robot is closer to the object and if the size of the object is not within desired frame, movements can be adjusted)

For communication, Bluetooth connection via listening to a specific channel of the device connected was complicating the service ports registered. Instead, we have resorted to using the server-client socket method by using one static channel for Bluetooth communication.

# 2) Image recognition

# 3) Android development

# 4) Algorithm

## **Implementation of the algorithms**

### **Hamiltonian Path implementation**

Since android app send the coordinates of the obstacles, a variable is created to store the list of obstacles with coordinates and directions. However, the distance between the obstacles and the robot are still unknown. Therefore, distance function is created to calculate the approximate distances between each obstacle.

A picture containing logo

Description automatically generated

Text

Description automatically generatedText

Description automatically generatedThe distances of obstacles and number of obstacles are known. For easier calculation, the graph function is created as below.

Text

Description automatically generatedUsing graph and distance functions, the vertexes and edge with weights are added by iterating through a list. Then, using the nearest neighbour algorithm, the path function will return a list of sorted obstacles. The implemented function is shown below.

Text

Description automatically generated

### **Path navigation implementation**

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated3 functions for turning are created for easier integration with robot movement. To keep track of robot facing direction, the turning angle is added once the function is called. The coefficient of pi is marked as theta hence 0 as east, 0.5 as north, 1 as west and 1.5 as south.

Graphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generated16 movement functions for the robot are implemented as the flowchart mentioned in log report 1. Few samples of the functions are shown below.

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedThe main function for path navigation algorithm (for obstacle at top right) is attached below.

Chart, scatter chart

Description automatically generatedGraphical user interface, table

Description automatically generatedIt is difficult to check whether the algorithm is working or not. Therefore, a simulator for robot is created to test out the algorithm. GUI is showned below.

## **Unexpected issues**

Forward function is iterated according to the calculated distance. The issue occurs once the result is negative. This issue is solved by calling reverse function. Another issue is that the algo does not check whether the obstacle is in front or not. The robot moves as the created move functions even if there is an obstacle in front. It is solved by creating one function to check the moving path is clear or not. If it is not clear, robot will move out to the free space and check again till there is no obstacle.

## **Implementation strategies**

The flow of the algorithms is thoroughly analysed in the design stage. A few functions are created in the purpose of reusing during the implementation to improve modularity and optimization. The comments are also added to improve the readability.