Mobile Robot localization challenge

Group 9: **Zixiang Lu**
s1818027@ed.ac.uk

I. Abstract

We have implemented a particle filter based localization robot using IR sensors for range measurement and hall sensor for odometry measurements. The robot was able to navigate itself in an indoor environment with walls and obstacles, and find Point of Interests (POI) marked by reflected tap. However the robot could not turn accurately to the target ("satellite") after it lost its orientation somewhere in an open space of the map, due to the range limit of the IR sensors.

II. Introduction

The task for this project is to run a robot in a known simple indoor space with walls and obstacles (Figure 1), then detect and stop on multiple unknown reflected tap Point of Interest (POI) area in the arena, and align the antenna on the robot to point at a known target ("satellite") on the ceiling. Since limited electronic and mechanical hardware are given to construct the robot, the task was attempted to achieve by using localization technics to obtain the position of a POI and do geometrical calculation with respect to the POI position and target position, then turn and point the antenna towards the target. Taking into account the imperfect performance of the given electronic components, a particle filter based localization method was attempted for minimizing the uncertainty of the robot produced by the motion and sensor noise. Using two IR sensors to measure the environment, one hall sensor to measure the robot's odometry and turning, and two light sensors to detect the reflective POI, the robot combined all these sensor information and scan around the arena with a pre-defined fixed path to detect the POI, record its location and turn to point at the target. Since the group was build up with members that has different previous backgrounds (Computer Science, Electronic, and Physics), the entire task was broke down into different minor tasks and each group member focused on each of their minor task to enable a higher efficiency for doing this project and take advantage of different talents of different members. As I am in electronic background, the task assigned to me was to test and integrate all the sensors that was being used then enable the odometry, turning and IR sensor measurements, as well as achieve the detection of POI. This report will demonstrate the methodology to achieve the task with focus on my own part, and it will present the results collected as well as a discussion summarizing the results.

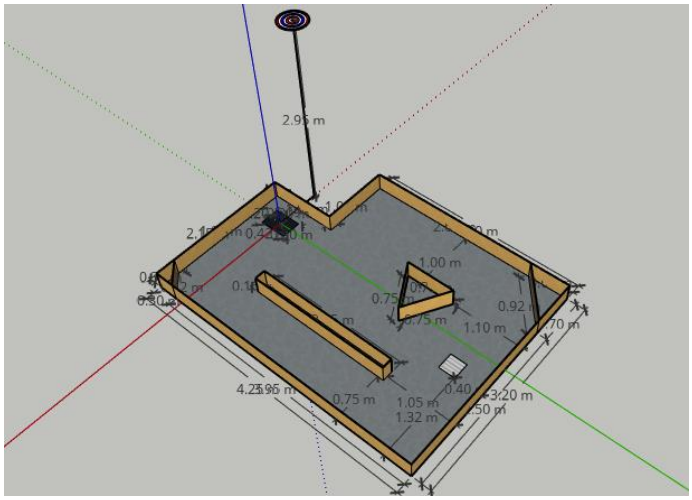

Figure 1, Map of the arena. Dimension is approximately 4.25m X 3.20m. The target ("satellite") is on the ceiling.

III. Methods

General solution

To achieve the task, one possible direct solution is to use a camera with image recognition techniques to detect POIs and the target, then stop on the POI and point towards the target. But taken into account that the computation power of the given processor is low and the environment could seriously affect image recognition precision, a simpler and more robust solution is to use the geometrical method to calculate the required angle of turning that could finally points to the target. With this method, the location of the POI should be obtained. Since the location of POI is unknown, the robot should continually keep track of its location and record that location after stopping on a POI. So this challenge of pointing to the satellite has been converted to the challenge of obtaining an accurate location of the robot, and the precision of the satellite alignment highly relies on the precision of localization.

Robot architecture

Initially, it was intended to construct a three wheel robot with two controlled wheels and one omnidirectional wheel at the back, because of the flexibility of movement and simplicity of design. However, after testing on the first three wheel robot architecture, it had been found out that the robot is difficult to travel in a straight line. As only one hall sensor was given to measure the odometry and turning angle, it is obvious that both controlled wheels should always be in the same velocity so the robot have to move as straight as possible. So the architecture was then modified to a four wheel robot (Figure 2) to gain more stability of movement. Thus the movement of the robot was constrained to be straight movement and rotatory turning. Additionally, to enable a stable and easy-

to-measure motion, the velocity of the robot was reduced to be as slow as possible, and this benefited the odometry measurement and a more straightforward on and off control. The gear ratio was then selected as 5:1.

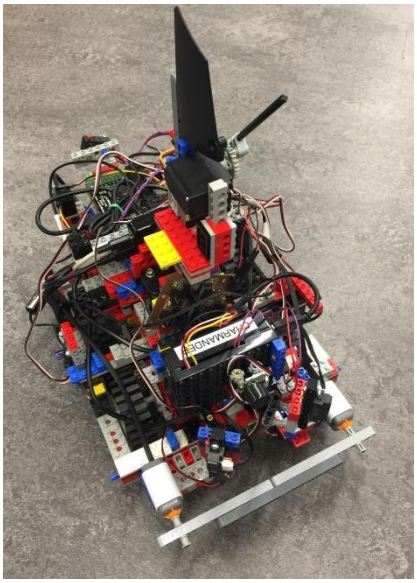

Figure 2, Image of the robot

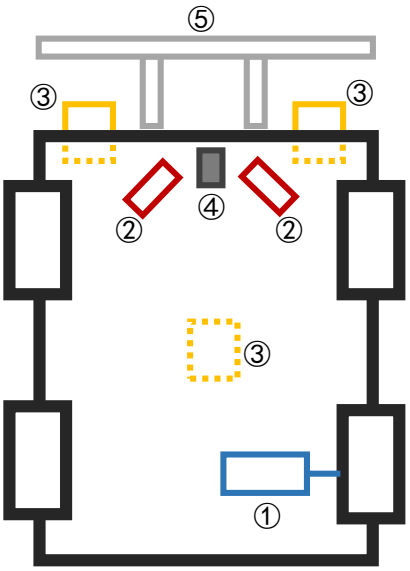

Figure 3, Sensor layout on the robot

Sensor integration

Five different kinds of sensors was used to measure the internal and external data as well as to avoid obstacles. The layout of the sensor integration is shown in Figure 3:

① Hall sensor:
1 attached at the back right wheel to measure the odometry and turning angle

② IR sensor:
2 located at the front of the robot to measure the range to obstacles with an angle of 45 degree to cover most of the space at the front

③ Light sensor:
2 at the front for detecting the existence of POI and 1 in the middle for stopping the robot on the POI

④ Sonar:
1 in the middle of two IR sensors to avoid obstacles

⑤ Bumper:
2 at the front to join a big bumper to avoid bumping into walls that is neglected by the sonar

POI detection

The detection of POI was achieved by the simplest way that uses the light sensor to detect the lighting difference between the reflected tape and the normal floor. To achieve detecting and stopping on the middle of the POI in a robust way, three light sensors was used. Two front light sensors was to detect the existence of POI with a rough direction (i.e. POI is on the front left of the robot if only left sensor detects a lighting difference), and one middle light sensor was to assist the robot to stop on the middle of the POI.

After testing on the real POI, the decision boundary of whether a reflective tape area is detected was a digital output value of 50 from the light sensor. If the light sensor output is less than 50 it means no reflective tape detects and vice versa. Taking into account that the robot might not always head towards the center of POI (in most cases the robot was usually just touch a corner of the POI), the output from both of the front sensors was used to detect the rough direction of the detected POI with respect to the robot itself. Therefore, there are four possible combination of the decision and there interpretation was listed into Table 1. [*In the notation of the following tables, "POI_false" represents POI is not detected, "POI_right" represents the POI is on the right of the robot, "POI_left" represents the POI is on the left, and "POI_true" means POI is underneath the robot.*]

Table 1, Left and right light sensor interpretation

Left light sensor digital value	**Right light sensor digital value**	**Interpretation**
< 50	< 50	POI_false
< 50	> 50	POI_right
> 50	< 50	POI_left
> 50	> 50	POI_true

As mentioned, the robot very likely to detect only a portion of the POI that could possibly miss the POI if the robot does not turn towards the center of POI after the detection. So an improvement had been made to the normal POI detection with a logic to turn towards the center of POI in terms of the status of the robot. Basically, the robot will refer to the previous state of itself and combine the current state to make a decision for its action. Since there are four possible interpretation of the two light sensor values, there could be 16 possible combinations of the previous and current state of the robot. The most robust way is to list out all the possibilities to a table and import this state table to the robot's state estimator. The combination of all the 16 states and there corresponding action to be performed is summarized in Table 2.

Table 2, State table of the logic for heading towards the center of POI

State		Action
S1 (previous)	S2 (current)	
POI_false	POI_false	Move forward
POI_false	POI_right	Turn right 2 counts
POI_false	POI_left	Turn left 2 counts
POI_false	POI_true	Move forward
POI_right	POI_false	Turn left 1 count
POI_right	POI_right	Turn right 1 count
POI_right	POI_left	Turn left 1 count
POI_right	POI_true	Move forward
POI_left	POI_false	Turn right 1 count
POI_left	POI_right	Turn right 1 count
POI_left	POI_left	Turn left 1 count
POI_left	POI_true	Move forward
POI_true	POI_false	Stop 30s
POI_true	POI_right	Stop 30s
POI_true	POI_left	Stop 30s
POI_true	POI_true	Move forward

Localization technique

The major work of particle filter location was done by another group member, but the basic description is demonstrated below. As explained above, the robot should have a good localization accuracy. The alternative localization methods that could produce accurate localization is grid localization and particle filter based localization. Initially the grid localization was attempted due to its simplicity of the algorithm. However, when actually implementing the grid localization, the difficulty is modeling the sensor and motion in the discretized grid map and the grid size could be very large if a higher resolution needed which would potentially consume too much computational power. Therefore, a more precise localization using particle filter is then attempted to be implemented.

Generally the robot utilities the sensor and motion information to update the randomly generated particles and select the possible particles to represent the real location of the robot with considering the noises of movement and sensing. Since the given raspberry pi has a computation limit and the task is restricted in 5 minutes, the update of particles was not attempted in real time to gain a faster speed. Instead, the particles were updated less frequently only on a number of pre-defined update points (or waypoints) in the map. The robot always try to reach the next waypoint based on its current location, so the path of

the robot could be specified by the order and the location of each waypoint in the map. The example particle filter localization method is shown in Figure 4.

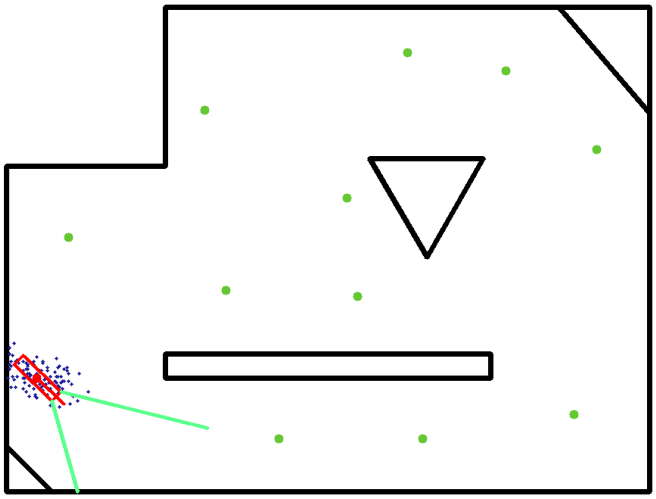

Figure 4, Example particle filter localization map. The red rectangle represents the robot. The green lines represent the IR measurement. The blue dots represent the particles location. The green dots represent the waypoints

The procedure of the robot doing the particle filter localization is demonstrated below. First the robot start at a known position and orientation. Then the robot will turn to the direction towards the first waypoint and attempt to move straight to that waypoint. After the movement the robot location is less certain due to the imperfect motion, but the sensor data is used to minimize this uncertainty by selecting the most possible particles in this state. The uncertainty of the location is reflected by the dispersion of the particles. The predicted location was taken as the mean location of a batch of possible particles. For the next step, the robot will turn towards the next waypoint and move straightly to that point, then another update of the particles will be performed to obtain the location. Repeat these procedures until the robot reaches all the waypoints that cover the entire map, the robot is able to scan around the arena with localization to detect the location of the POIs and point the antenna with the correct angle. Example procedure of localization is shown in Figure 5.

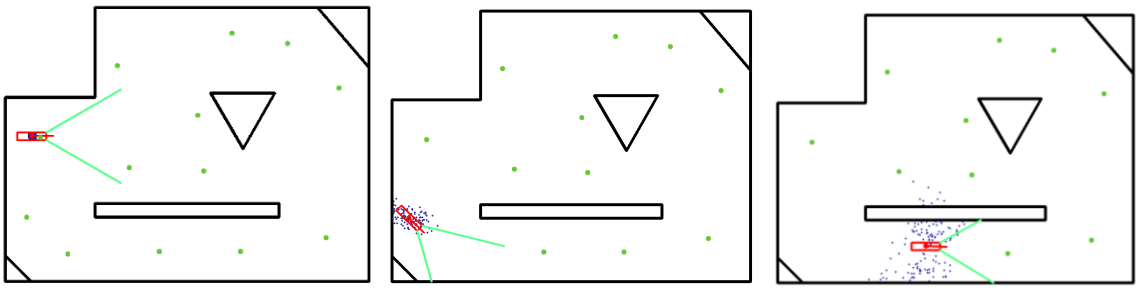

Figure 5, Procedure of the localization in the task

Sensor Measurement

Hall sensor

To measure the odometry and turning angle of the robot, the only selection is the hall sensor since using a time counter to map out this measurements are not consistent and unreliable. As only one hall sensor allowed, the robot's motor speed of all wheels should be maintain the same to keep the measurement independent on left and right wheels. Basically, the hall sensor could produce identical square waves after the shaft of the hall sensor spins. So it can measure the rotatory angle of the motor by counting the number of square waves produced. This could be simply achieved by detecting the edges of the square waves and iterate a counter after detecting an edge. The edges of the square wave could be detect by checking the difference between the previous and current output. If the previous and current output are different then an edge is detected and vice versa. The pseudo code for this part is shown in the Algorithm 1.

Algorithm 1: Hall_counter

Input: None
Output: displacement D, turning angle φ
Initialize: counter $C = 0$, current hall sensor output S_t, previous hall sensor output S_{t-1}, output difference ΔS
S_t = *get_hall_input()*
$\Delta S = S_t - S_{t-1}$
$S_{t-1} = S_t$
if ΔS is not 0 then
    C += 1
$D = 1.5C + 1.3$
$\varphi = 7.5C + 1.7$
Return D, φ

IR sensor

The IR are the only solid range measurement unit for this task, as the performance of the provided sonar is extremely terrible that both the resolution and reaction speed are not sufficient to obtain the information from the environment. The output from the IR sensor are Analog to Digital conversion (ADC) values that could not be directly used as the range measurement. An interpretation from this digital value should be convert to the actual values. This was performed by testing the digital outputs with respect to the real distance measurement to the wall of the arena, and fit a curve that best describes the numerical data then do a pseudo reverse to obtain the characteristic function of the IR sensor.

IV. Results

Hall sensor

With the method demonstrated above, the hall sensor could measure the rotatory angle of the wheels. To obtain a higher resolution, the gear ratio between the wheels and hall sensor is 1:8 using a gear increasing technic to enable faster rotation of the hall sensor to detect more edges. After integrating the hall sensor onto the robot, the displacement and turning angle performance was tested using actual measurements. The plots of displacement and turning angle versus the number of hall sensor counts are included in Figure 6. After obtaining the characteristic of the hall sensors, a linear regression was applied on the results.

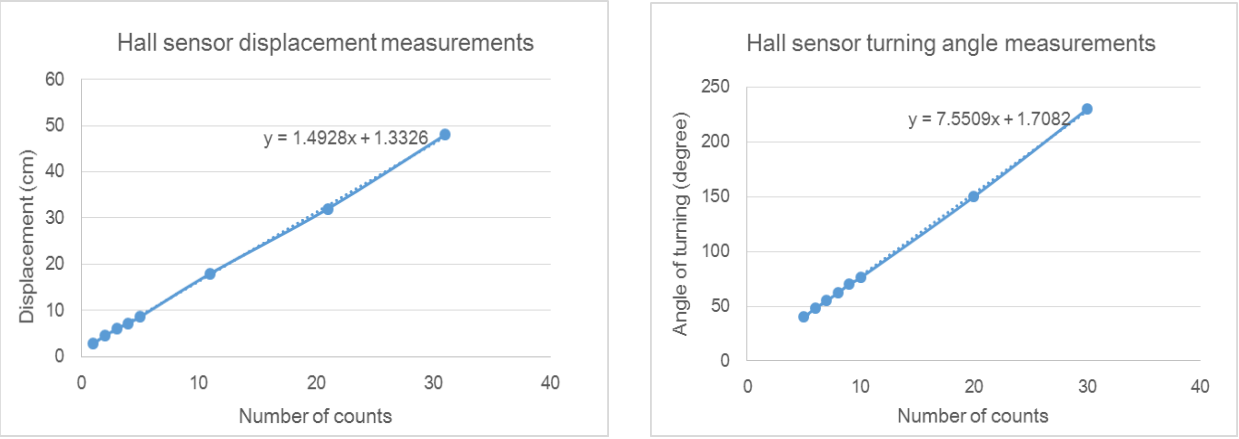

Figure 6, Hall sensor measurement characteristics

As shown from the above results, both displacement and turning angle measurements are very linear. This supports the robot with a consistent measurement of itself. After fitting a linear regression, the resulting equation was obtained as *displacement = 1.5*hall_counts + 1.3* and *angle = 7.6*hall_count + 1.7.* This results show that the resolution of the robot is 1.5cm for displacement and 7.6 degree for turning angle. This resolution is good enough for achieving this task taking into account the error allowance.

IR sensor

The output from IR sensors are not directly the range to obstacles but a digital value converted by the analog signal of the sensor. So the IR sensor should be tested for obtaining a numerical representation of this sensor's characteristic. To experiment on this sensor, identical interval of actual ranges with a step of 10cm was selected and at each range 100 samples of the digital output was collected to construct a dataset, since the IR sensor tend to have a higher dispersion of the outputs especially at a large range. Multiple curve fitting technics (polynomial and logarithm) was attempted and the results along with the original data in Figure 7.

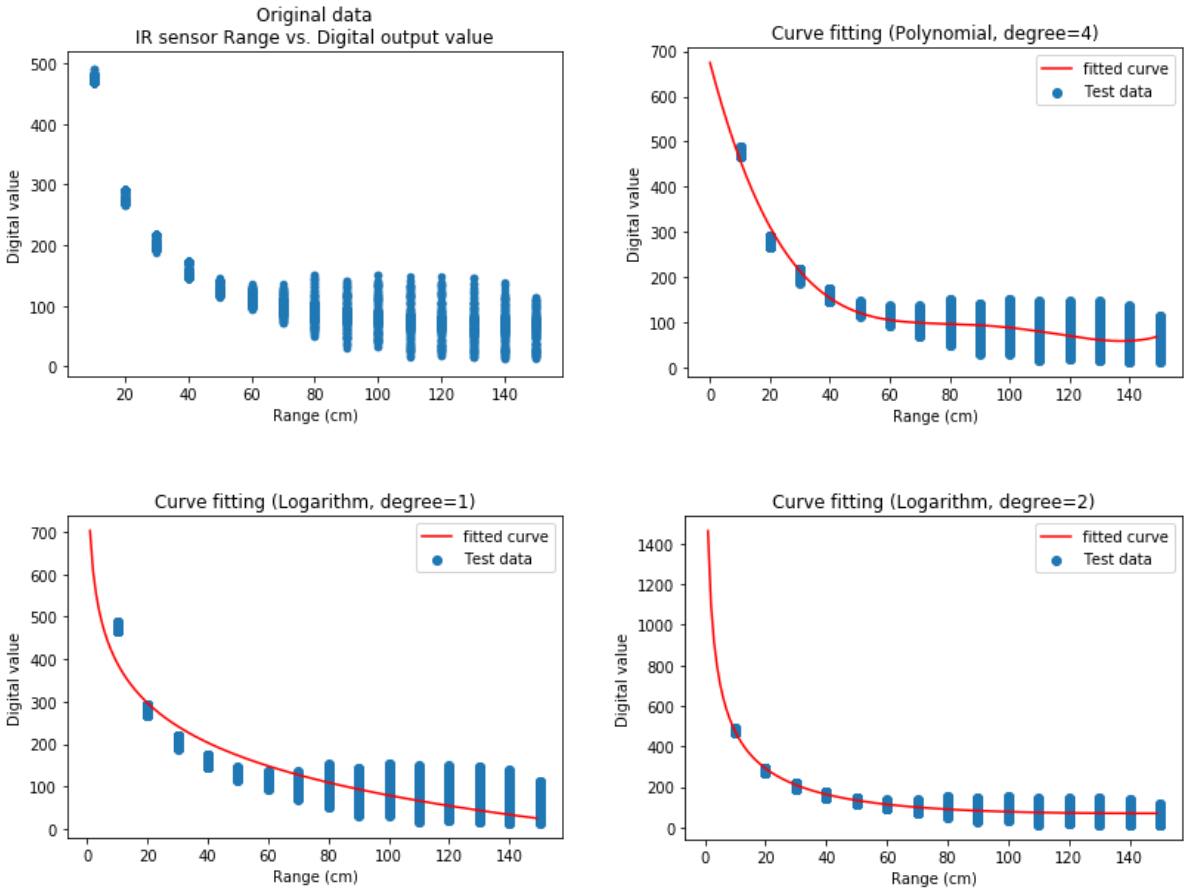

Figure 7, IR sensor raw measurements and multiple curve fitting methods

From the results above, there are significant noise of the IR sensor and this noise increases as the range increases. The standard deviation of the output increases significantly after 70cm and maintain the same. Visualizing the shape of the raw data, the curve is possibly a log function. After fitting by a 4 degree polynomial function, a 1 degree logarithm and 2 degree logarithm, it is apparently that the 2 degree logarithm best fits the dataset.

The fitted function of the data:
4 degree polynomial: $y = 7.5 \cdot 10^{-6}x^4 - 2.9 \cdot 10^{-3}x^3 + 4.1 \cdot 10^{-1}x^2 - 25.2x + 674.5$
1 degree logarithm: $y = -135.456\log(x) + 702.868$
2 degree logarithm: $y = 56.214\log(x)^2 - 559.988\log(x) + 1463.919$

However, when looking at the numerical results, the reverse function of the 2 degree logarithm is relatively complex, although it produces the best fit. So for simplicity, the 1 degree logarithm was used to represent the data since its reversing function is the easiest one. Testing on the robot, the IR sensor could produce 90% accuracy of the output.

The final milestone task

The robot was able to perform localization with particle filter updating at each waypoint, and real time display the location of the robot in the map. The robot was able to produce good localization when it was in a narrow space which both IR sensors can correctly detect walls. However, when the robot head into an open space that only one IR sensor or none of the IR sensors worked, the robot will lost itself especially giving incorrect orientation that affects the pointing antenna terribly. In the final milestone, the robot is able to detect all POIs, but it did not point to the satellite on the first POI and it pointed to incorrect direction to the satellite at the second and third POI because it lost its orientation somewhere around an open space.

V. Discussion

In summary, this robot could perform perfect movement and good internal and external measurements with robust POI detection and accurate pointing towards the satellite at a given location and orientation. However, there is still many problems in the particle filter localization and some of the problems (e.g. losing location and orientation at open spaces) are fatal. The most successful part I did was to improve the resolution of the hall sensor to a high level that most of other groups could not achieve, and the enhancement of POI detection that the robot was able to head towards the center of POI with only a touch at the boundary of POI. The system could be possibly improved by testing on different layout of IR sensors, and to fine tune the motion and sensing noise as well as the waypoints parameters.