University of BRISTOL

Course: COMS0012 Robotics Systems

Coursework: CW2 Real Robot Localisation

Group: POLH-2

Members:
Yizhang Peng (yp16505@my.bristol.ac.uk),
Zilin Lu (zl16533@my.bristol.ac.uk),
Haiping Hu (hh16860@my.bristol.ac.uk),
Kazeem Onisarotu (ko16613@my.bristol.ac.uk)

Task Breakdown:

Name	Percentage	Task
Yizhang	30%	Robot design, algorithm design
Zilin	30%	Algorithm improvement and test
Haiping	20%	Hardware, Documenting, Discussion
Kazeem	20%	Report writing, data recording

Robot Hardware Design

For the design of the robot, the team decided to go for a simple and minimalistic design from the provided parts using only an ultrasound sensor and 2 motors with built-in odometry sensors. The Mindstorms "brick" is placed at the centre of both wheels of the robot and a castor wheel attached to the tail to support the structure.
The ultrasound sensor is placed at the middle of the two wheels, which is the center of turning in order to simplify the robot modeling and also placed at an appropriate height for detecting the walls.

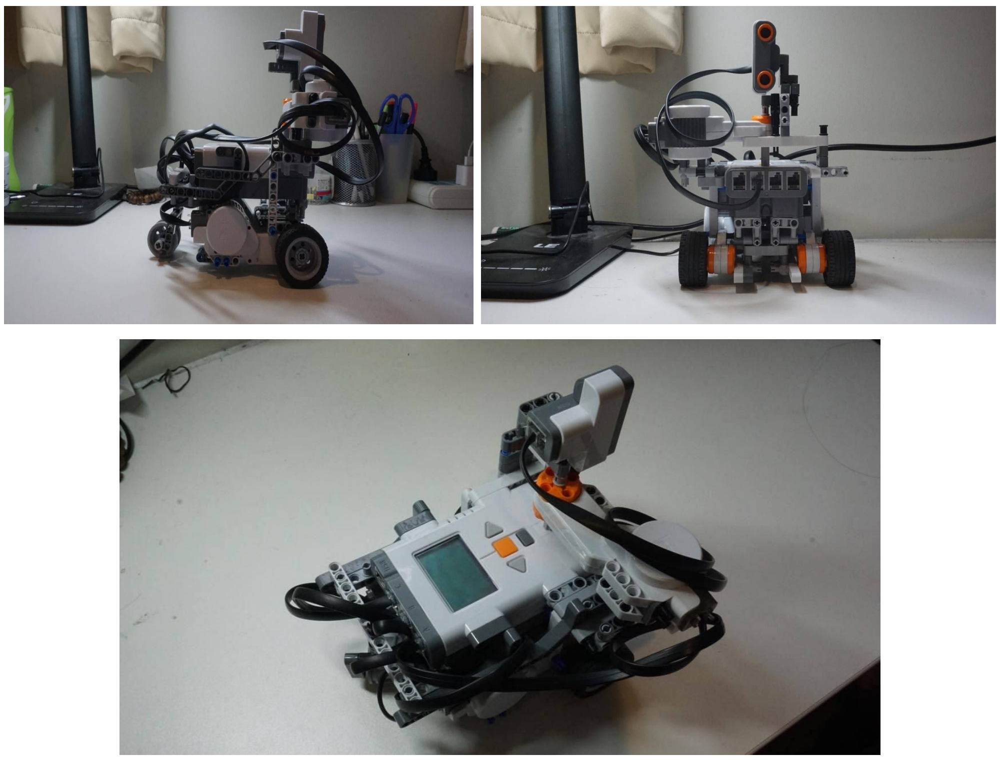

Figure 1 picture of the robot

Software Implementation

Real_robot Functions

At the start of project, the team decided to create library functions to control movement and sensors of the robot. These functions were implemented to provide ease of access and efficiency when creating the main program.
The library functions are listed and explained below:

1. Move_direct: This function is used to decide the movement of the robot either

forward or backward to any distance in cm, by setting the output power of the motors to the same value.

2. Turn_sensor: With the ultrasound sensor connected to a motor of the robot, this function is used to rotate the sensor left or right to any degree to allow for optimal scanning
3. Turn_sensor_back: This function simply rotates the sensor back to its original starting position after it has been turned.
4. Ultra_scan: This function is used to implement the scanning performed by the robot, it uses the turn sensor function to turn the sensor while it takes readings of nearby obstacles.
5. Pre_turn: This function is used to calibrate the robot's initial turning angle to the best preferred angle by taking 18 initial scans and using the retrieved data to find that angle.
6. EvaluatePath: This function takes the turning angles and the distances of a path analyzed by the robot and evaluates the path by summing and returning similar distances with a set turning angle.
7. Final_path: This function takes a path and analyzes it by taking both positive and negative angles into consideration and finding their differences to get a final distance, it returns a final path with the turning angles and distances. This function was devised for efficiency and to speed up movement towards the target path.

Localisation Strategy

For the localisation strategy, a particle filter was created with 600 particles randomly distributed around the map at random locations and orientations to represent a hypothesis of the state of the robot.

To avoid the inaccuracy of the return value from the sensor when the robot is at a skew angle, the pre_turn function was created to make sure that the robot can start with zero degree. With the robot at its starting position and its best starting angle (zero degree), an ultrasound scan is performed 4 times (90 degree turn each time) to retrieve information about the robot's surroundings, the particles also perform the same number of scans using BotSim's ultrascan method and the value for each scan is scored to provide a weight for each particle.

On the next phase of motion prediction of the real robot, we encountered the issue of the robot hitting the walls, so we decided to solve this problem by setting a threshold on the minimum distance to an obstacle when the robot scans. If the minimum distance scanned is below that threshold, the value of some of the distances retrieved from the ultrasound sensor are changed while the minimum distance is set to zero. Another threshold is set and used to decide the robot's next move, with the robots preferred movement to orientate towards the scanned maximum distance to an obstacle and move directly towards it but never the entire distance to ensure it never hits that obstacle, otherwise the robot just turns and changes its orientation but never moves.

As the robot changes its orientation by turning and moving, each particle will perform the exact the same movement done by the real robot. Next was the resampling of the particles according to the weights of the current particle, the first 120 particles were resampled and assigned new positions in accordance to weight, while the remaining particles were assigned new random positions inside the map.
To check for the convergence of the particles, the position of each particle is retrieved and sorted in ascending order, after the sorting has been completed, for a certain number of particles the mean of the x coordinates and y coordinates is calculated and used to estimate the x and y coordinate of the real robot.

Path Planning Strategy

After the localisation of the particles to formulate a probable position and state of the robot, the next step was to set a plan that strategizes a path that enables the robot to move from its current location to its target position. With the map initialized and the estimated coordinates of the current position received from the localisation strategy, we used wavefront planner algorithm to find the path towards the target. The algorithm for the path planning strategy has been divided into three parts, including map initialisation, border widening and and the actual path planning.

Map Initialisation

The map was divided into several rows and columns, the distance between each row and column is 5 cm to ensure many crossing points in the map. When the map is initialised, the crossing points which are not in the map will be given a value of 0, while the points that are in the map will be initialised according to their distance to the target. The closer the distance to the target, the lower the value.

Border Widening

In this part the of the algorithm, the border of the map will simply be widened. This will sharply increase the cost value of those points which are close to the walls, in order to prevent the robot including them in its path towards the target.

Path Planning

With the target position known, wavefront planner algorithm is implemented and used for the path planning. Using the values from the distances from crossing points to the target location, the robot will tend to favour movement from the larger values towards the smaller values. Since the points are initialised using a matlab map array, we had to implement certain issues that the real world robot could enter during movement towards the goal. In the case where an obstacle was in the way of the robot and its target point, a threshold was set to ensure that the robot never hits an obstacle and after it has approached the obstacle, the robot simply re-orientates and turns angles and then continues to move towards the target position. Using the evaluate path function, the robot analyzes the chosen path to the target location and evaluates it by by taking all similar distances and summing them and setting it to the best angle, the path is then examined again using the final path function to take into consideration the effect of positive and negative angles, it takes the distances of the positive and negative angles and sums them up and returns a final path to the target point to be used by the robot.

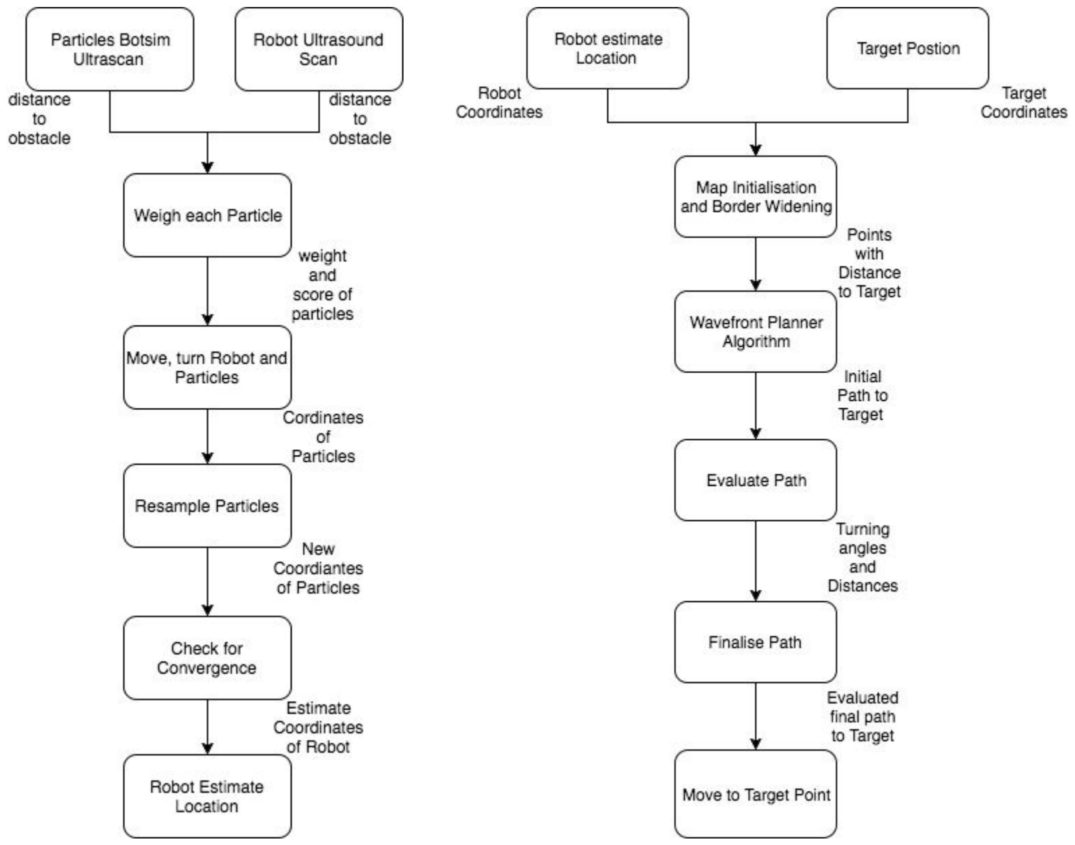

Figure 2 the flow diagram of the algorithm: localization with particle filter (left) and path planning (right)

Discussion

Difficulties Overcome

Over the course of this project, the following difficulties and issues were encountered.

Sensor Error

During the testing, the ultrasound sensor proved to have a very low accuracy when the incident angle was more than 30°. If the robot performs a high resolution scan, it is very likely that the sensor measures an incorrect value. To prevent this problem, the resolution of the scans was set to 4, trying to be vertical to the walls. However, this strategy also made the robot very to have a very limited localization ability when it is placed in the room with a random angle to the wall.

Robot Initial Angle Detection

To solve the problem raised in the last part, an angle detection strategy was applied to ensure the robot is rightly placed in the room before the localization process. To initialise and calibrate the robot it performs an initial 18 scans in 270°, the data retrieved from the scan is then sorted in ascending order in accordance to distance to get the shortest distance. If the difference between the shortest distance and its two successive distances are greater than or less than a set threshold, the data retrieved is used to calibrate the robot to best starting angle and helps to solve the issue of if the robot is placed in non zero angle.

Obstacle Avoidance

To avoid the robot driving into the corners of the map, the robot will detect the distance between itself and the wall and if that distance is smaller than a threshold, the value of some of the distances retrieved from the ultrasound sensor which are adjacent to the shortest distance will be changed and the minimum distance is set to zero. In this case, the robot will not go to that direction and hence the robot will not hit the corners and walls as well.

Improvement In Speed

To increase the speed of the whole process, several strategies have been applied.

Scanning

The motor which rotates the sensor turns to a different direction each time. As a result, the sensor doesn’t need to be reset to the starting point each time, by simply using the endpoint of the last scan as the new starting point.

Combining each path point

The path plan was originally recorded as a set of checkpoints, a previous was implemented where the robot was to be moved step by step to follow each point on the path. The improved method now combines all the points on the same line together thereby reducing the number of the checkpoints and creating a more continuous movement of the robot.

Motor Speed

The motor could get a more accurate angle when it was set to a low speed, but improving the accuracy of the motor may use more time. So the team formulate a strategy that adjusted the speed of the motor to gain the best performance.
In the localisation strategy, the speed of the motor should be set low since the accuracy of localisation is of high importance while in the path planning stage, the robot only needs to move towards the target hence the speed will be increased. The team decided on a final strategy where the speed of the motor is decreased in the localisation strategy to gain more accuracy and increased in the path planning strategy.

Potential Improvements

We may improve the algorithm we used to increase the speed and accuracy of localisation. In addition, we also can shorten the path from the robot to the target.