CS-684 project

[Team -17]

Exploring unseen arena using swarm robots

Satyabrata Behera [113050064] (leader)

Hrushikesh Mohapatra [113050002]

Toshendra Sharma [113050013]

V.G.S.Neela Lohith [113050047]

Requirements

Functional requirements

- The project is about exploring the unseen arena using small, cheap robots which are sent into the arena.
- A map representing the unseen arena constructed by receving the signals sent by swarm robots.
- The input for the matlab is the signals received from the bots using XBEE (Based on zigbee protocol, IEEE 802.15.4) which gives information about coordinates.
- These coordinates are mapped onto the matrix using matlab.

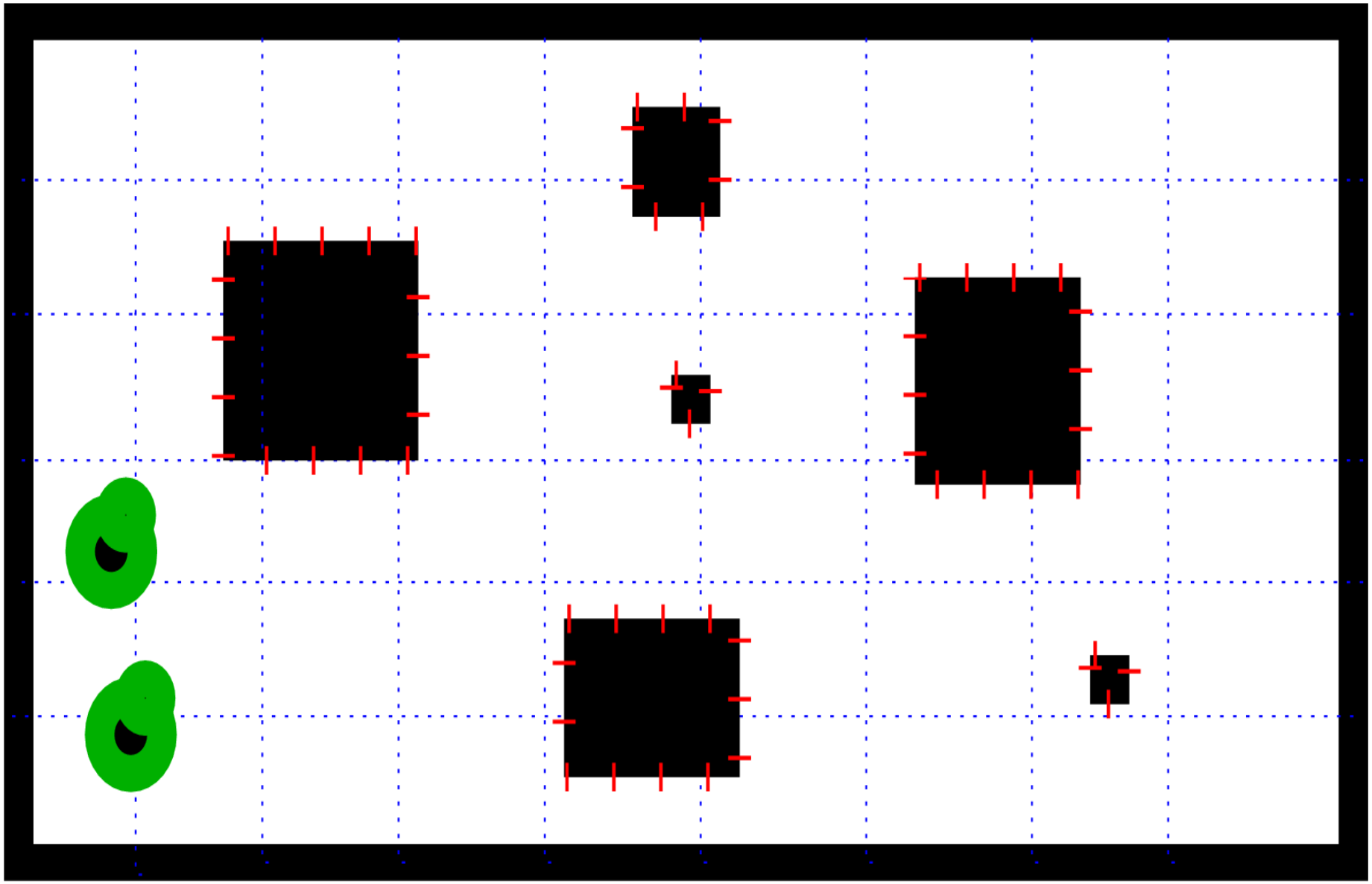
Problem Statement

- Multiple robots which are cheap, unintelligent and random moving, also called swarm robots are released into an unseen arena. Each and every bot keeps sending the distance they have traversed and the angle of rotation, either when they meet obstacle or when the wheel does a complete rotation.

- The signals are sent using Zig-bee protocol. The matlab interprets these signals and converts them into coordinates by using polar coordinates and plots them onto the image.

- After some duration(which Is inversely proportional to the number of bots used), the image contains points where there are no obstacles and a distinguishable boundary of the obstacle.

Proposed Arena



Requirement Specification

- The proposed specification includes swarm robots moving randomly in all directions and algorithm embedded in these bots send the distance traversed and the angle rotated when they come across an obstacle.
- These parameters are taken as input by the algorithm in the matlab and polar coordinates are calculated and are plotted on the image.
- After certain duration we get rough boundary of the obstacles on the image and there by we can identify these obstacles in the unseen arena.

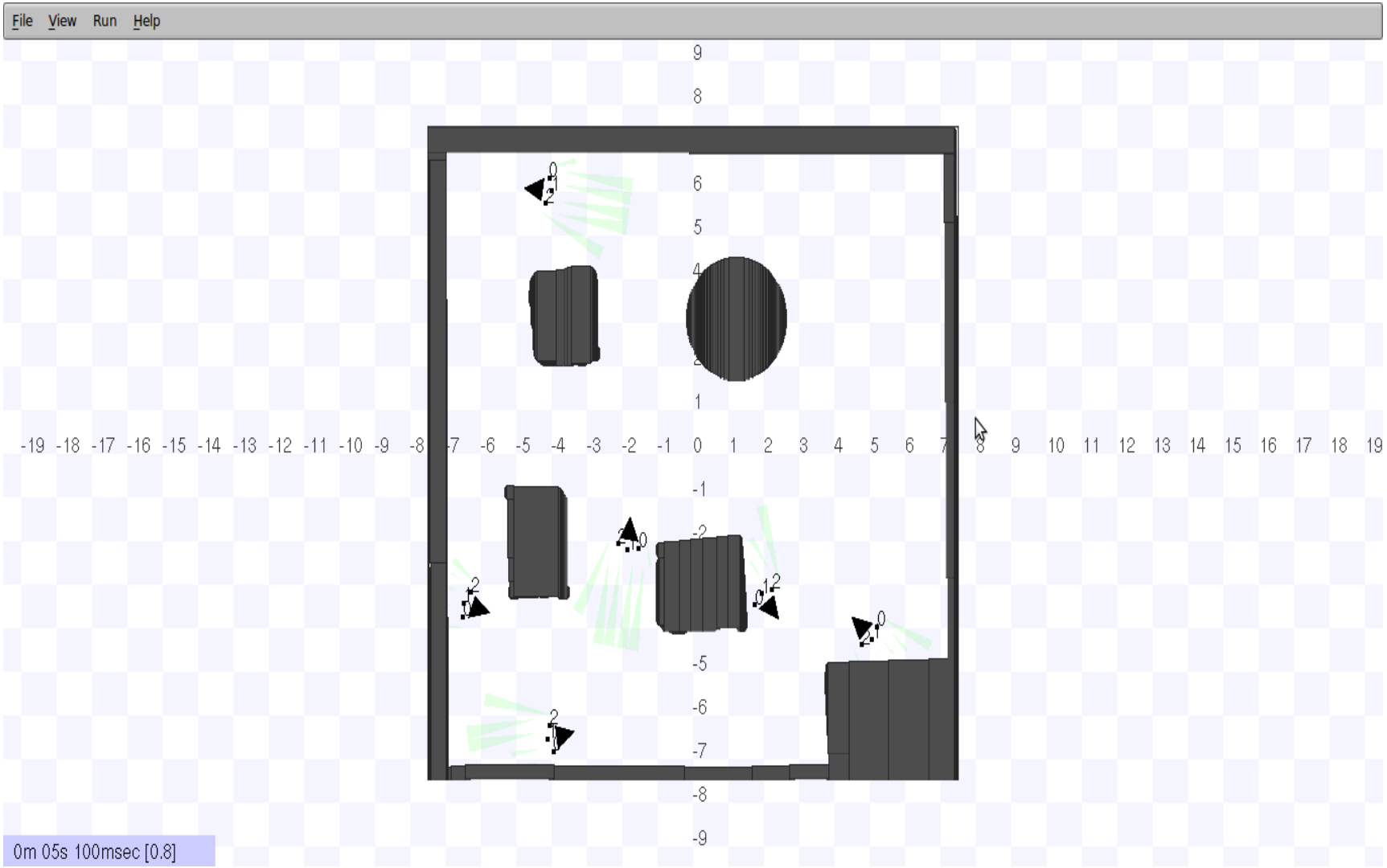
Final System

- Since there may not be many collisions, we modified the algorithm so that we get continuous signals from the bots and zigbee module to matlab and a reasonable amount continuous points can be seen plotted on the arena, where there are no obstacles.
- If all robots send signals continuously, it becomes difficult for matlab to handle simultaneous multiple signals. Hence the bot sends signals only when they come across an obstacle or when one of the wheels take a complete rotation.
- Algorithm is scalable for a huge number of robots. Hence, scalability factor is efficiently implemented.
- Due to errors in the hardware, mat lab image isnt analogous to the path traversed by the robot.

Continued..

- To verify the validity of the algorithm and the logic we came up with, we simulated the entire project using Stage.
- In stage simulation we have put 6 instances of the robots, with 3-4 obstacles. Each robot moves randomly in all directions and each possessing 5 sensors.
- Hence each robot treats each other as the obstacle too.
- Every coordinate of each and every robot is stored in the file and after around 80-100sec, all these points are taken and plotted using GNUPLOT.

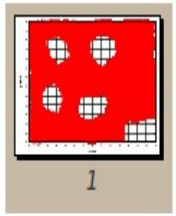
Simulation results



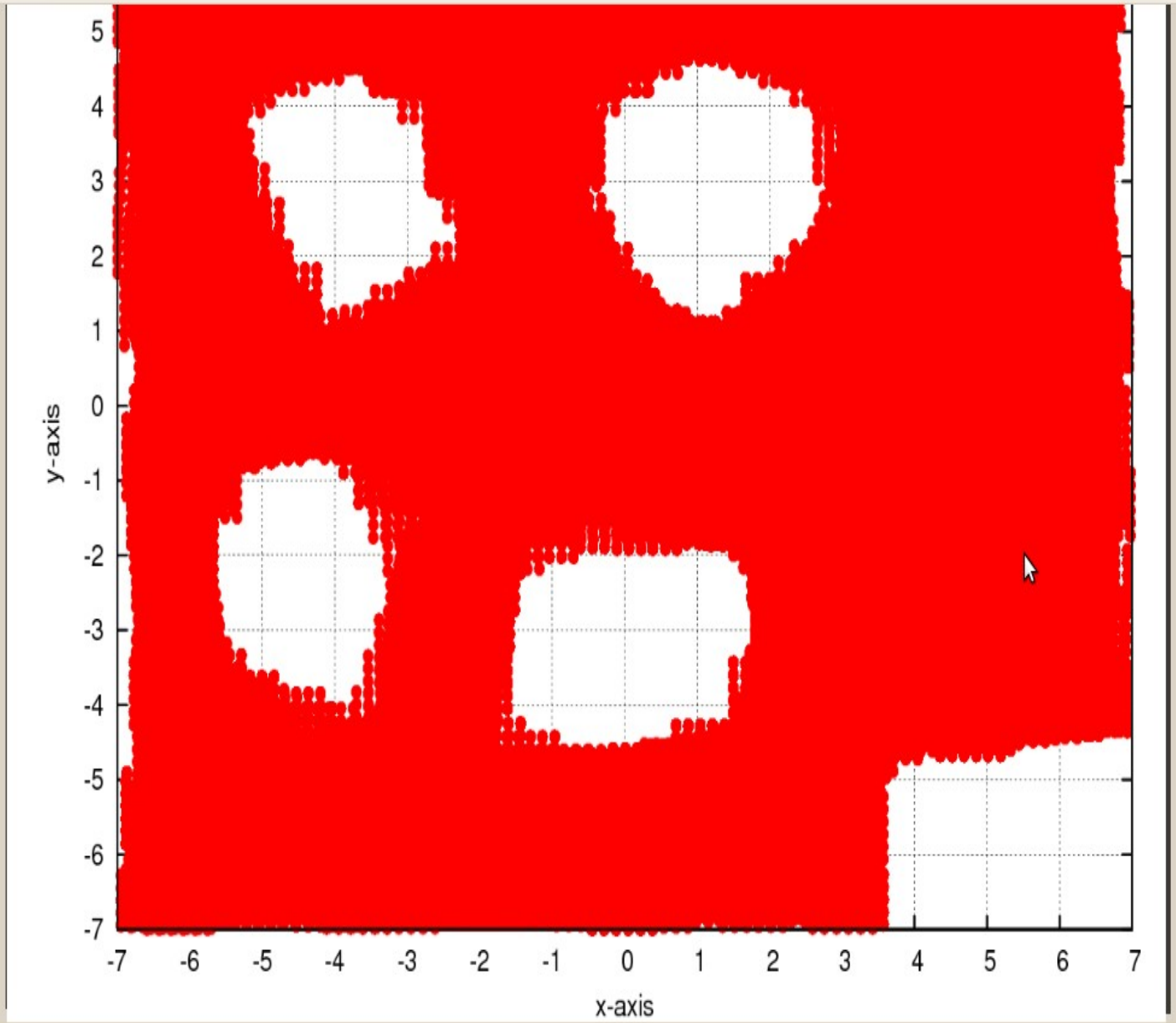
File Edit View Go Help

Previous Next 1 of 1 200%

Thumbnails



1



Problems faced

The spark V robot doesn't possess obstacle detection sensors, but IR proximity sensors. Hence it becomes mandatory to cover the walls of the arena boundary and the obstacles with white surface.

Further, swarm robots need to detect the each other in the arena as the obstacle. Since Spark V is not covered with a white surface, it has become difficult to place more than one robot on the arena.

There is an error with the wheel-encoder. Initially we were under the impression that the error in calculating the coordinates could be due to slip in wheels. But we started to face problem with the variation in velocity of the two motors.

To overcome the above problem, we tried to calculate the difference in the velocity of the two motors and hard coded by reducing the velocity of one wheel with the difference calculated.

But it dint seem to work, as the variation in the velocity is not constant through out.

The other solution we tried out was to reduce compensate the variation in the velocity by reducing/increasing the velocity dynamically by using an algorithm which measures the difference in the shaft counts of left and right wheels and base on that value, decision is taken and velocity of the wheel which is moving at a faster pace is reduced.

But this algorithm wasn't able to solve the problem because of the error in the shaft count values.

Bot may take 45 degree turn intitially and algo needs to compensate that turn by moving -90 degrees, else deviates the path.

But it is not feasible because the bot then keeps moving in a zig-zag path and hence the distance traversed by the bot is way more than what it needs to traverse, thus results in wrong results. Further, The algorithm is only used to DETECT the error but not PREVENT it, it becomes infeasible to get the coordinates with accuracy.

In Spark V, one shaft count value = 12.85 degrees. Hence, when an obstacle is encountered, distance traveled and the previous angle rotated is sent as the input to mat-lab. Hence each time an obstacle is encountered, there is a an error of 6.5 degrees on an average. Thus, when the bot is given an instruction to rotated at an angle of 102 degrees, it may rotate 90 degrees and since, we are calculating the cumulative distance, the error keeps incrementing and the bot deviates from the path which is displayed by the image.

Future Work

- Scalability is important. Our project can now be scaled to around 999 swarm robots.
- Boundary following robots.
- Triangulation method can be used over wheel-encoders which cause many errors.
- Robots for house hold applications such as house cleaning robots can be made and also used for military applications
- A camera can be mounted on the robot to get all details of the arena.

Thank You