Swarm robotics documentation

Tejas Chikoti

June 2021

1 Introduction

The obstacle avoidance consists of three steps

- Detecting obstacles
- Localisation so that it avoids these obstacles
- Globalisation so that it reaches the destination in least time

2 Setup and basics

2.1 pseudo-code

```
While(EverythingIsOk)
{ bool b1=RayCastFromObjectToDestination();
  If(b1)
  {GetAsCloseAsPossibleToTheFirstObjectDetected();
   AvoidTheObstacle();}
  else
  {GoToTheGoalInTheDirection();}
}
```

2.2 TrainableParams

- Goal-Velocity-This Determines how fast you will be proceeding to the Goal
- Target-Velocity-This will determine how fast the object will be shifting behaviours from Localisation to Globalisation.
- Threshold-This will determine how close the object can get to a particular obstacle
- Noise-This will determine how much noise should be added to the Localisation algorithm(As there are many directions for the object to go avoiding the nearby obstacle)

2.3 LidarParams

- RayRange-How far the LIDAR can shoot a Ray to for localisation (although I assumed the use of LIDAR or any big scale instrument for Globalisation.)
- NumberOfRays-In practice it is Infinite but i have assumed a finite number.
- Angle-This can be termed as the field of view(In case of Rotating Lidar it can be made to 360 degrees).

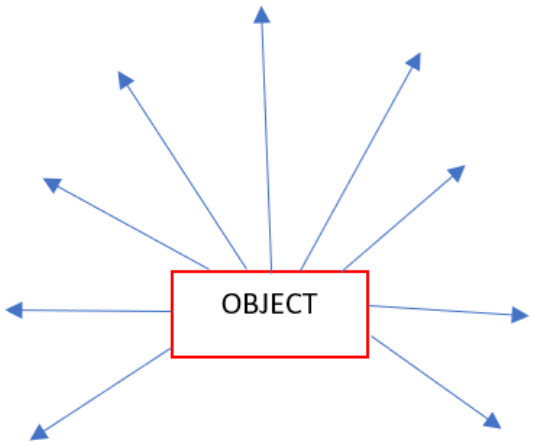
2.4 Inputs

You will have to give the Inputs in terms of the final X coordinate,Y coordinate,Z coordinate this can be related to the GPS globalised system like latitude and longitude for large scale but this algorithm won't be applicable till there.

3 Application

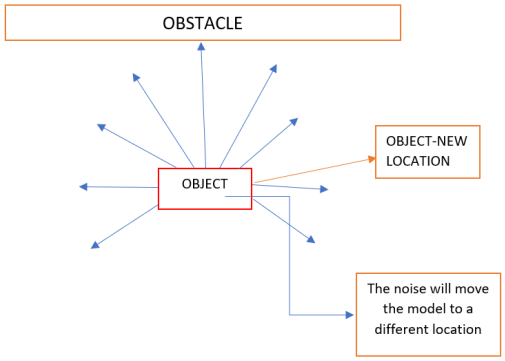
3.1 Localisation

The localisation phase begins with sending rays in various directions covering the Field of view as shown in the Figure.Now these Rays return bool values whether it has hit an obstacle or not Now using this information we move in the direction in which rays have not been hit and then continue in the direction it was moving previously.



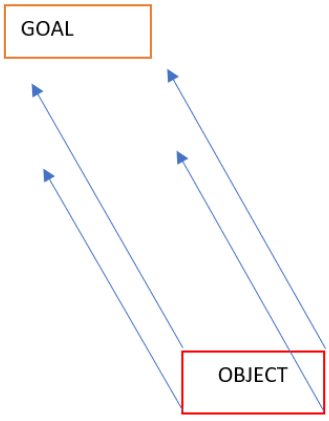
3.2 Noise

Now it is crucial to add noise for the localisation model as for the above algorithm to work we require it to be having at least one ray for which it doesn't detect the obstacle or else it will remain there only why we add noise of sinusoidal in nature as it has a constant amplitude and this can maybe free it from the situation(at least we expect that!!).



3.3 Globalisation

We send four rays from the each of the corners of the object with Goal distance in the direction to the final destination and evaluates the ray and checks which ray has detected an obstacle and we store the least distance for which the object has hit and the object moves in that direction until it less than the threshold distance and then the control switches to the localisation.



4 Improvements I have done

- I have added noise to the original model i saw in the references
- I have added the globalisation so that the object reaches its destination.
- I have added Thresh-hold so that the object doesn't get too close to the obstacle.

5 Advantages And Drawbacks

5.1 Advantages

- This can work with irregular objects like sphere ellipsoid etc.
- The threshold feature ensures no damage to the bot.
- The localisation module implemented here has a large number of trainable parameters and can be experimented with according to your object's need these largely depend on the obstacle size and the bot size.(Like you can use a different noise if you want instead of sinusoidal etc).

5.2 Drawbacks

- for large distances i have used the Ray-cast but this may not work for large environments such as factories.(i.e. globalisation module.)
- This model can only work in an infinite space environment like for instance it may not work in the a floor that has finite breadth and length and down of it is pure air.

6 Potential Fixes

- Like we can make a variable target velocity and goal velocity and optimize the time complexity further.
- Further the noise can also be variable and can only be implemented when there is an issue like the bot is stuck there for say 30 frames
- Maybe we can change the globalisation module to with other path planning algorithm like Bug1,A* and introduce noise there also.

7 References

For localisation `https://www.youtube.com/watch?v=SVazwHyfB7g&t=1315s`