International Core Journal of Engineering Volume 7 Issue 5, 2021
ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

Simple Improvement of Braitenberg Vehicles

Jieyu Fan
Twug Wah Senior High School, Dongguan, Guangdong, 523000, China.
fandra77@vertical-china.com

Abstract

Simple Improvement of the Braitenberg Vehicles, which aims at making the movements of simple car models more adaptive, is made by utilizing Object detector, which is a tool in Matlab to detect the position of the car and provides the position information for the algorithm to complete this process. In a word, to make the car turn by itself and based on the position information provided by Object detector in order to avoid all obstacles is the main goal of this project. Also, the comparison will be made among all the results from different versions of algorithms to finalize the algorithm, which is capable of making the car move in the fastest and most smooth ways while avoiding all obstacles. For further simulation of algorithm, my project will try to design more routes in more complex obstacles' environments--triangular routes with avoiding three obstacles.

Keywords

Adaptive; Object Detector; Avoiding All Obstacles; Algorithm Comparison; Complex Environments.

1. Introduction

1.1 Motivation:

When I learned about Braitenberg Vehicles , I became interested in it because It has a simple structure and easy to understand. I regarded it as a reaction like Phototaxis (But different direction), which means a specie reacts to the strength of the light [1]. This gave me an idea about that if a car can have a sensor which can distinguish different sources to achieve a similar way of how those species work. This sensor can help this car move without hitting the obstacles in the environment. Achieving complex movements can be made by an addition to a simple structure. This really attracts me to achieve it because it is relatively easy to understand for a zero-basis student like me. So this is why I want to have a try in this project.

And this project resemblance with how neurons work. receiving a signal and reacting to it. This is the field which I am most interested in . So this bring me a lot of motivation to do research about this project. Maybe it is too easy for further study. But I hope this is the beginning of my study.

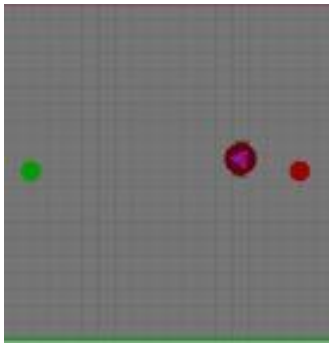

Figure 1. The obstacles' environment (2 sources)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

1.2 Goal:

My goal aims at simulating movement like moving towards 2 (Figure 1) and 3 sources (Figure 2). 3 sources are placed triangularly.Designing the suitable paths for different number of sources.

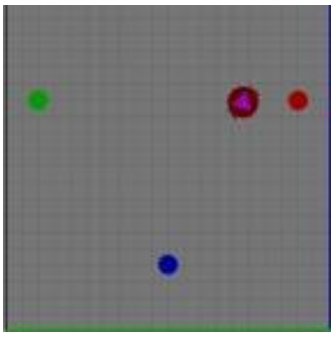

Figure 2. The obstacles' environment (3 sources)

1.3 Problems:

The biggest problems are that it is difficult to distinguish different sources and design the suitable paths for different situations. The former one is caused by one single sensor cannot provide the position information of the entire environment for judging the time for moving in the programmed way or reacting to the sudden change to the environment because the lack of differences among these information. And designing suitable paths for moving towards 2 and 3 sources needs a lot of time to try and determine that which path is the most fast and convenient .

1.4 Hypothesis:

To resolve the problems, Based on the theory of Braitenberg Vehicles, I think using the object detector (Figure 3) to detect the position of different sources and then making it move in a pre-programmed way when it does not move too close to one of the sources can solve the problems of direction.

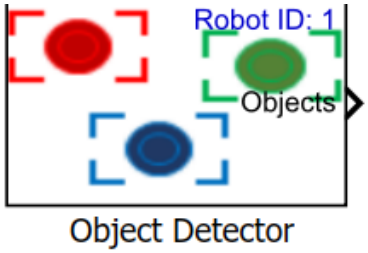

Figure 3. Object Detector (in Matlab)

Simply, the object detector furnishes the other sensors with preconditions of the actuation. The other sensors can only be used before there is not enough spare for them to turn.

With the information provided by the object detector, it gives us a possibility to analyze the information and run the simulation at the same time. While correcting mistakes and error made in the simulation with the diagram offered by scope (Figure 4), we can also clearly record all the simulation's results and routes, even all the position information which will be used in the comparison part.

Figure 4. Scope (in Matlab)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

1.5 Proof:

To prove whether my algorithm is successful,using my algorithm in a fixed environment, comparing the actual routes with the pre-programmed routes to find problems and differences which can help to improve the algorithm. For different environment , I need to design suitable routes for each of them. And run the simulation to record the routes the car goes through and improve the pre-programmed routes to be suitable.

Like comparing triangular orbit for three sources with circle orbit for three sources, which orbit can make the shorter path, the shorter time of reaction and the shorter safe distance form the source which the car is moving towards.

1.6 Implementation Plan:

1.6.1. Tools: Matlab , Robotics Playground (Figure 5)

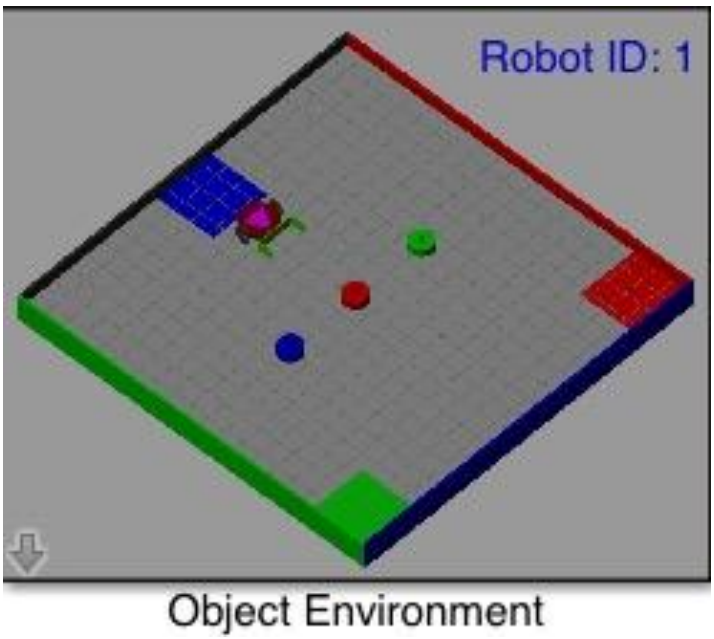

Figure 5. Robotics Playground (Object Environment Part)

1.6.2. Methods:

Step 1. Simulate the algorithm in the Robotics Playground

Step 2. Record the results

Step 3. Compare with the former result (If there is not,comparing it with the ideal result, which means the most convenient fastest route)--->

Improve the algorithm by correcting the mistakes and make a plan of the next version

Step 4. Simulate the new algorithm and repeat this process.

1.6.3. How to Compare the results:

Method 1. The cost of the time

Method 2. The times the car hits the sources

Method 3. The simplicity of the algorithm

2. Methods Model and Algorithm

2.1 Methods:

Step 1. Using object detector to provide the position information of objects to determine whether it will turn to move towards an object or just moving forward.This can help the car adapt to a new environment more easily because it does not demand a lot of revision of the program.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

Step 2. Using the limitation of the angle that the detector can detect to minimize the mistakes of judging the differences between different objects.And monitoring the car's angle so that it can turn any angle I want it to turn.

Step 3. Also I add additonal objects to make the environment more complex.And design its own special paths which is most suitable for completing one turn.

2.2 Models:

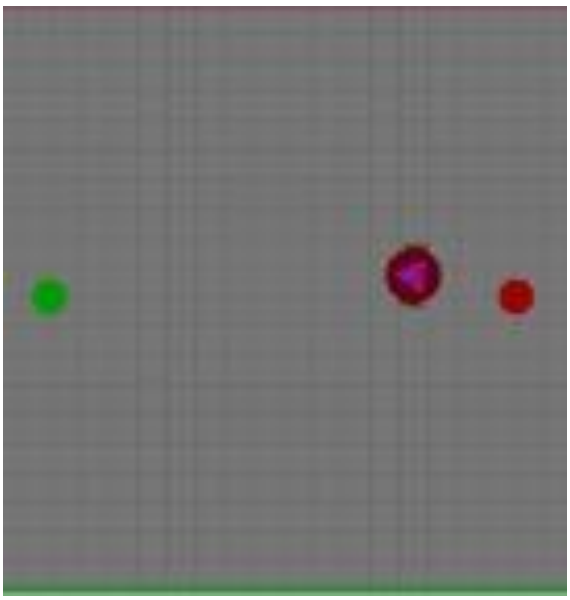

Figure 6. 2-object environment for path designing

Model 1. For 2-object environment (Figure 6), I design two simple paths for it. Their algorithms have something in common. They both need to turn around when they are moving too close to one of the object.So the difference between these two paths is that whether the car turns clockwisely (Figure 7) or anti-clockwisely (Figure 8) [2] when they move towards one object too close.

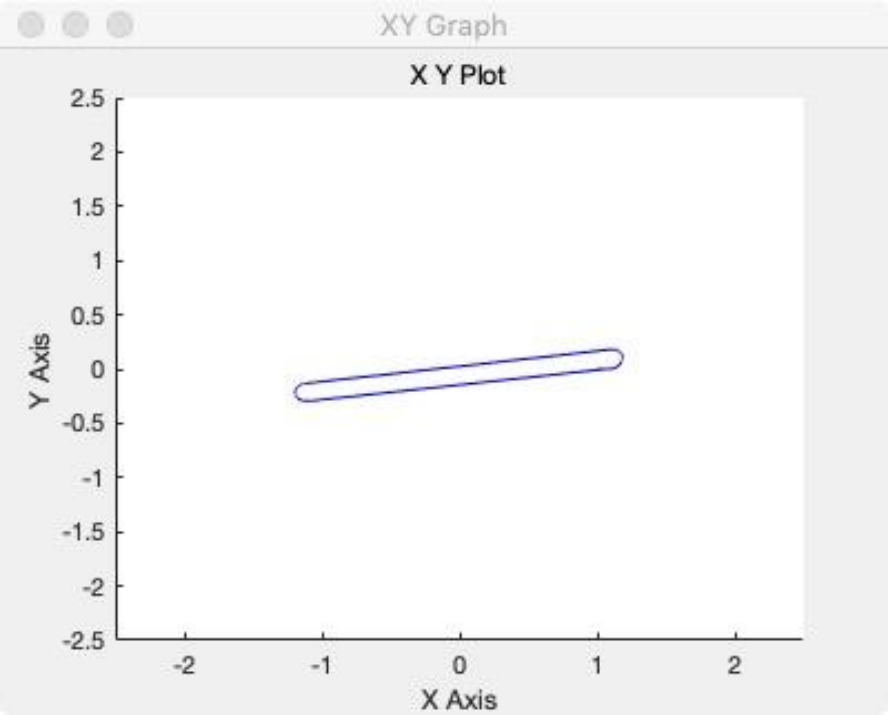

Figure 7. The movement of the car (Clockwisely)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

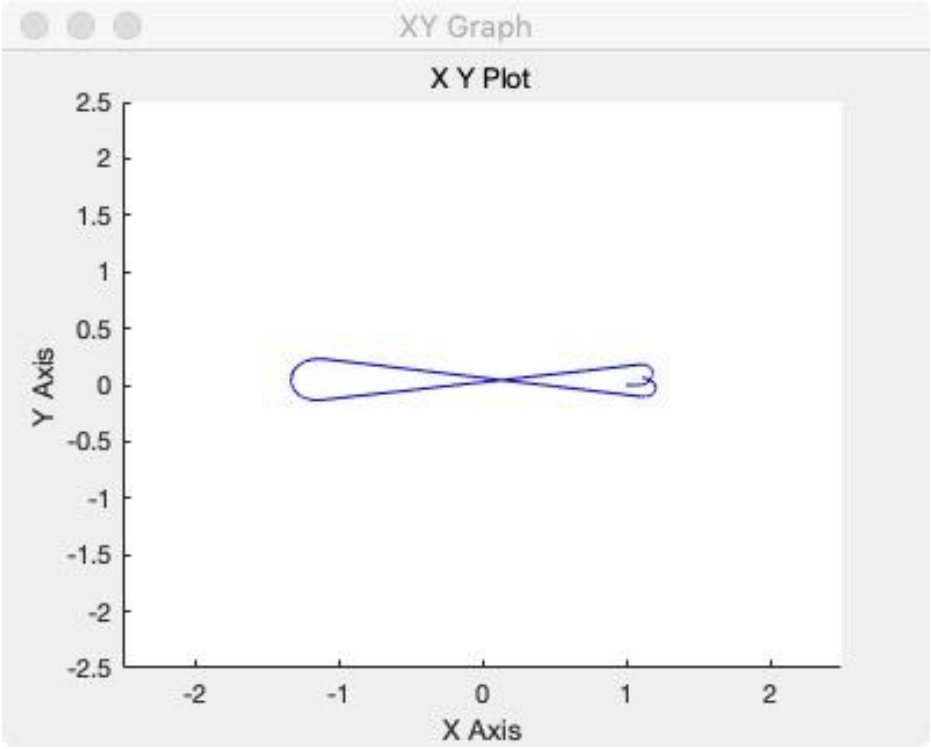

Figure 8. The movement of the car (Anti-clockwisely)

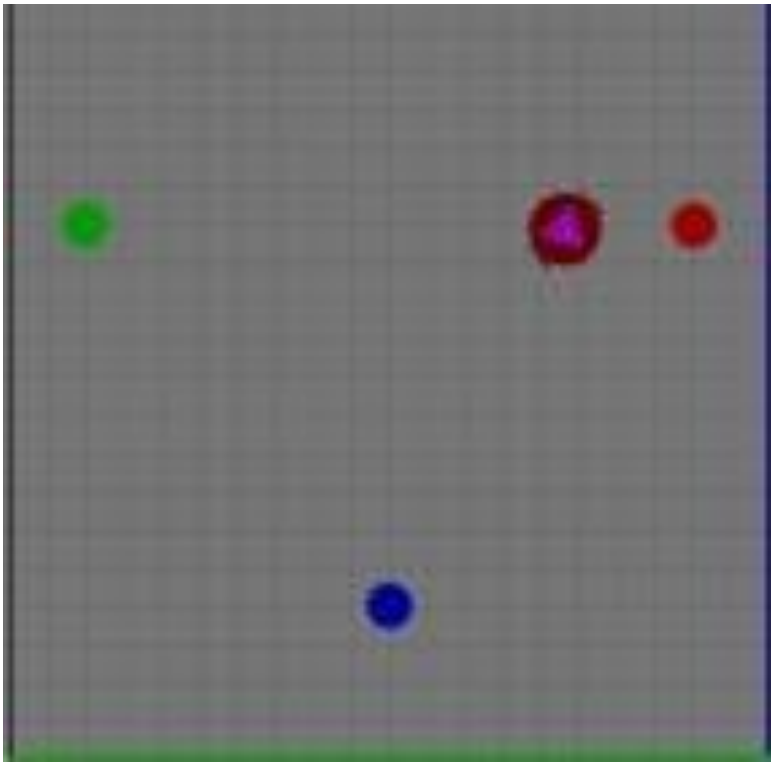

Figure 9. 3-object environment for path designing

Model 2. For 3-object environment (Figure 9), it will be a little more difficult to find the best way to calculate the angle it should turn to make the track complete. And the left picture below shows the former version before the revision. It cannot move continuously which cause this moving process cannot be finished after the first round (Figure 10). But after the calculation,I can finally make this process repeatedly (Figure 11).

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

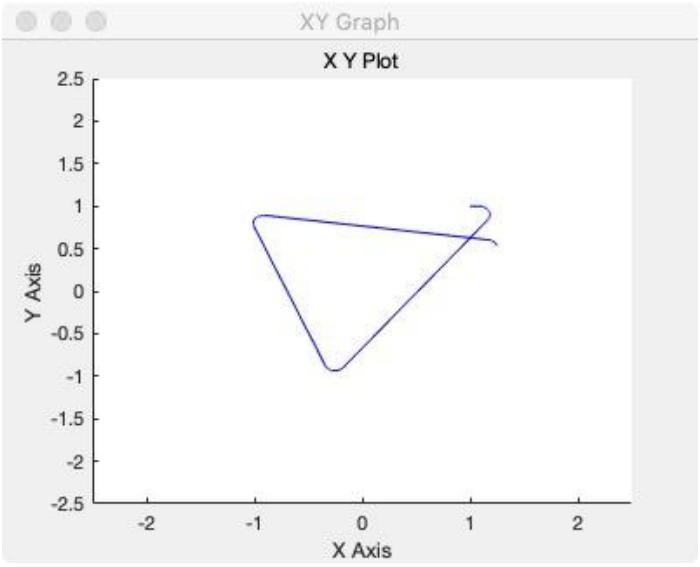

Figure 10. The movement of the car (After the first round around all sources)

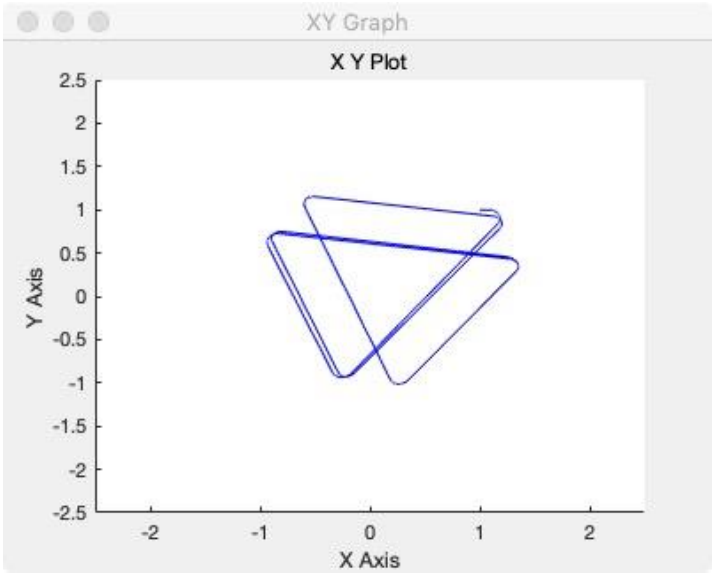

Figure 11. The movement of the car (Repeatedly)

2.3 Algorithm:

Feature 1. Using“if” for judging the time that the car needs to turn (Figure 12)

```
if dist1<1
status=1;
else-if dist2<1
status=2;
end
```

Figure 12. One part of the “if” algorithm

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

Feature 2. Using“status”to record the pre-programmed way used to navigate the car. (Figure 13)

```
switch status
case 1
    left=60;
    right=-40;
if abs(angle)<=4*pi/3
    status=0;
    left=60;
    right=60;
    return;
end
```

Figure 13. One part of the “status” algorithm

These are two main ways for this project. The former one is used for judging the time, while the later one is used to navigate the car about its movement when it moves close to one of the objects in the 2 and 3 objects environment.

3. Analyzing for Results

3.1 Situations Analyzing

In this project, we mainly analyze the results of the 3 objects environment.For illustration, we divide this process into three situations.

3.1.1. First situation: (Figure 14)

Figure 14. The car is turning when it face the red object and then moving towards the blue object.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

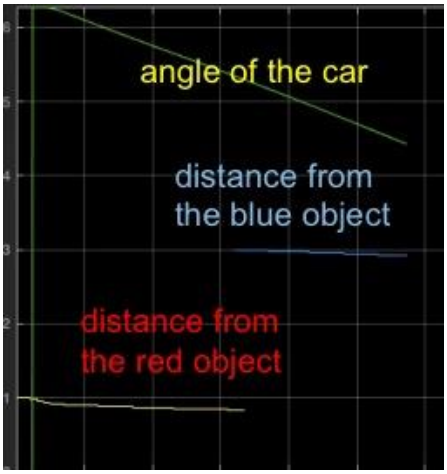

Figure 15. The data shown in scope (First situation)

For the data we get from the scope (Figure 15), we can clearly see that the distance from the red object (dist1) will not be detected when the angle of the car exceeds the limits of the detecting angle.

3.1.2. Second situation: (Figure 16)

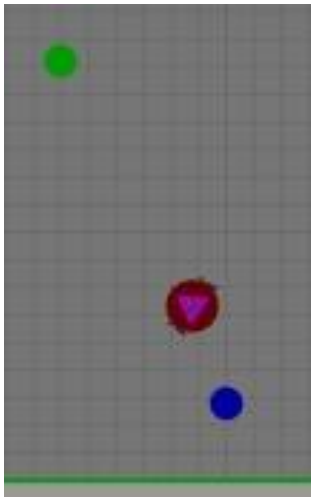

Figure 16. The car is turning when it face the blue object and then moving towards the green object.

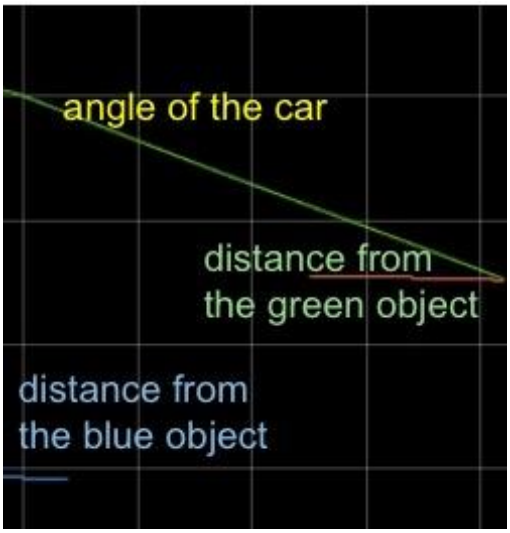

Figure 17. The data shown in scope (Second situation)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

For the data we get from the scope (Figure 17), we can clearly see that the distance from the blue object (dist2) will not be detected when the angle of the car exceeds the limits of the detecting angle.

3.1.3. Third situation: (Figure 18)

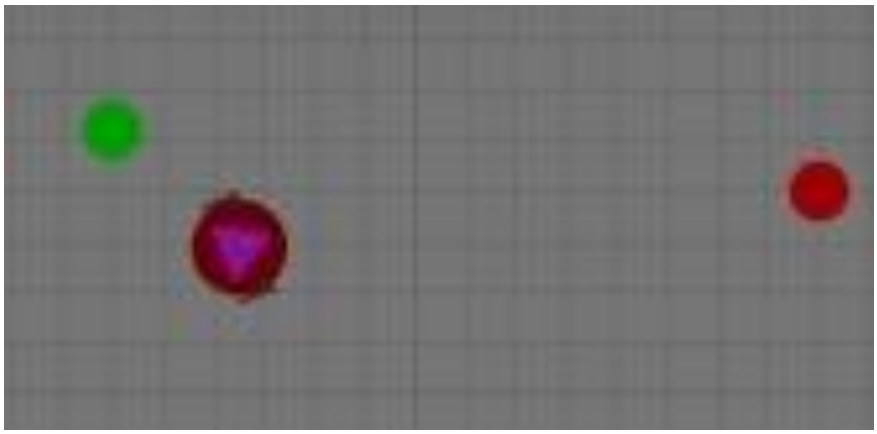

Figure 18. The car is turning when it face the green object and then moving towards the red object.

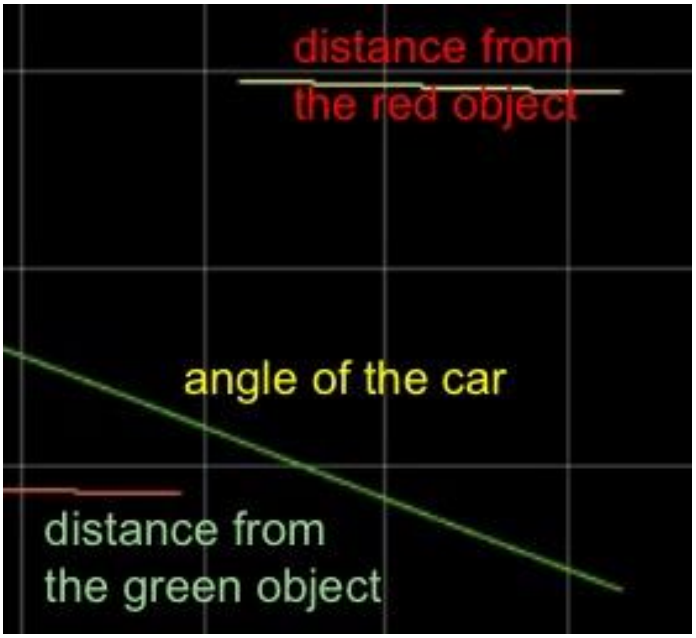

Figure 19. The data shown in scope (Third situation)

For the data we get from the scope (Figure 19), we can clearly see that the distance from the green object (dist3) will not be detected when the angle of the car exceeds the limits of the detecting angle.

4. Conclusion

With the position information provided by the object detector,it makes complex movement of the car possible. The car can achieve some basic reaction to obstacles without trouble since it can avoid all obstacles accurately.

Also, being adaptive to multiple situations (different number of obstacles) can be made by “case” algorithm.

All in all, two main goals of this projects are achieved without problems and errors. For further improvement of this project, I mainly aim at making the algorithm to control the car more automatically to different situations, by making a general algorithm which is suitable for all situations.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202105_7(5).0072

Acknowledgments

Adding the object detector as a sensor to provide the position information of the objects and take control of the movement of the car is feasible by using the Function in Matlab to give the signal of how the car can move. By comparing the differences between the program for 2 and 3 objects environment, I conclude that different situations need different algorithm for the movement of the car, like the calculation of the angles varies from 2 objects to 3 objects environment.

References

[1] https://en.wikipedia.org/wiki/Braitenberg_vehicle#External_links Mechanism Part.

[2] https://en.wikipedia.org/wiki/Braitenberg_vehicle#External_links Behavior Part.

[3] http://people.cs.uchicago.edu/~wiseman/vehicles/#Introduction Part.

[4] http://users.sussex.ac.uk/~christ/crs/kr-ist/lecx1a.html AI Lecture:Braitenberg Vehicles.