HORA 2019
www.horacongress.com

International Congress on Human-Computer Interaction, Optimization and Robotic Applications

July 5-7, 2019, Urgup, Nevşehir, Turkey

SETSCI Conference Proceedings
www.set-science.com
4(5): 55 - 60, 2019

A Car-Like Robotic Experimentation for Path Planning Study

Ahmet Emre Danışman and Tankut Acarman

Department of Computer Engineering, Galatasaray University, Turkey
aemredanisman@gmail.com, acarmant@gmail.com

Abstract – Scientific progress is an ongoing process and humans always have new ideas for new inventions. Although, scientific process is cumulative, you must invent the tire before inventing the car. Currently, mankind has past that point and looking towards to new challenges. One of the new ideas is cars that drive themselves, in a more formal and general way, the 'autonomous ground vehicles'. Autonomous vehicles in city traffic have been a dream for a long time. However, this great idea comes with its own problems. There are many parts of autonomous driving such as scene understanding, path planning. Many methods have been studied and developed to solve these problems.
In this paper, a novel path planning algorithm and results of extensive experimentation on it are presented. The experimentation were done in a custom environment consisting of a custom-built vehicle, a desktop computer, a camera and an open source marker library.

Keywords – path planning, autonomous ground vehicles, real-time computing, robotics

I. Introduction

One of the fundamental problems of creating an autonomous ground vehicle is the navigation. It can be defined as the set of maneuvers that enables the vehicle to move from one location to another safely. In order to call this navigation process intelligent, the autonomous ground vehicle should be respecting the environment and its surroundings during its movement towards the goal. This can be achieved by intelligent algorithms that take the obstacles and the capability of the autonomous ground vehicle into the account. So, it has been a research topic for quite some time and many different algorithms and approaches have been studied over time [1]. More information on brief history of autonomous ground vehicles and different fundamental algorithms can be found in the following sections.

II. Methodology

Path planning algorithms can be defined in a very simple way, they find a collision free path from an initial point to a final point. Simplicity of this definition is the ultimate sophistication that allows so many different approaches for path planning algorithms. Over the decades of collaborative study, there are some fundamental algorithms which are the basis for almost every other algorithm. These algorithms include, but not limited to; A* [2], probabilistic roadmap [3], rapidly exploring random tree [4] and potential field algorithms [5]. The algorithm proposed in this paper works real-time and is a novel algorithm that can be alternative to mentioned algorithms.

Our proposed algorithm has 2 steps. The first step is based on a derived version of Hybrid A*[6] and the second step is curve-fitting.

Hybrid A* algorithm is an improved version of well-known A* algorithm, it also takes a basic vehicle model into account while creating path. It does not generate any extremely sharp turns and the generated path is smoother in general. The first part of our algorithm adds up on Hybrid A* and makes improvements on it.

The second part of our algorithm treats each obstacle in the experimentation space that blocks the way separately. It fits curves around them to avoid collision and connects them with the path parts that the first part generates. Eventually, the path is produced minding the obstacles.

After the path is finalized, we direct the vehicle to follow it and make it possible with a custom controller. The controller controls the steering and torque. It calculates the steering using two parameters. It adjusts the steering value given by the path depending on these parameters. One parameter is the difference between the current steering value and the steering value that should the car have at that moment. The other parameter is the offset, the distance that vehicle has drifted away from the given path. After calculating the parameters, our controller calculates the new steering value and sends it to vehicle. The controller controls the torque to command the vehicle to go forward or backwards. Trajectory planner occasionally outputs paths including driving backwards and out vehicle is capable of doing it.

III. Experimental Study

In order to analyze the performance of our algorithm, we have done numerous experimentation with our custom setup. The main components of the setup are the vehicle, the desktop PC and the camera. The focus of the experimentation was testing the ability of feasible path generation and following the generated path as well as re-routing if needed. The main parameters of our experiment are vehicle's starting position, starting direction and obstacles on the field.

A. Experimentation Setup

We have built the vehicle from scratch using the LEGO Mindstorms NXT 2.0 set [7]. The vehicle is a 4-tire car sized

DOI: 10.36287/setsci.4.5.013

24 cm long, 17 cm wide 14 cm high. It has 3 electric motors on it, 2 of which powers rear-wheel driving and 1 control the Ackermann Steering. The motors are connected to the 'intelligent brick' that has a 32-bit ARM7 microprocessor [8]. The intelligent brick is connected to the desktop PC via Bluetooth. On top of the vehicle and on the ground, as obstacles, we have ArUco markers [9]. ArUco is an opensource OpenCV library that has unique markers with IDs and returns their position and orientation. Markers are used to detect the vehicle's and obstacles' positions and orientations.

B. Experimentation Scenarios

In this section, you can see screen outputs of some experimentation scenarios as well as a close-up look on the generated direction and the path followed. Not every single experimentation scenario was presented in this paper for the sake of simplicity and page constrains. There are two images on each screen output. The right one is the camera image of our experimentation environment. On the camera image, there is vehicle with a square marker on top of it and numerous square markers by themselves, representing obstacles. The left image is the output of our algorithm, which shows the obstacles, the vehicle and its orientation, the generated path (shown as blue line), and the followed path (shown as red line) as the vehicle moves forward. The last figure of each scenario shows the motion responses that are drawn using the logged data.

There are two obstacles on the bottom of the images in each scenario. Independent of the starting position and direction of the vehicle, the goal position and direction of the vehicle is between the obstacles and facing downward. In the Figure (a) of each scenario, the vehicle is at its initial position. In Figure (b), the vehicle is approximately halfway through the planned path. In Figure (c), the vehicle has completed its journey and arrived the goal. Figure (d) of each scenario shows a closer look at the experiment, drawn from using the logged data of the scenario.

All the conducted experimentation scenarios can be found in the table below.

Table 1. Experimentation Scenarios

Scenario Name	Vehicle Position	Vehicle Direction	Obstacles (Amount, Position)
Park01	Center	Straight	-
Park02	Right	Straight	-
Park03	Left	Straight	-
Park04	Center	Straight	1, Center
Park05	Right	Straight	1, Center
Park06	Left	Straight	1, Center
Park07	Center	Right	-
Park08	Left	Right	-
Park09	Center	Left	-
Park10	Right	Left	-
Park11	Left, Close	Right	1, Left
Park12	Right, Close	Left	1, Right
Park13	Left, Close	Straight	-
Park14	Right, Close	Straight	-
Park15	Center, Close	Reverse	-
Park16	Right, Close	Reverse	-
Park17	Left, Close	Reverse	-
Park18	Left, Close	Right	-
Park19	Right, Close	Left	-
Park20	Left, Close	Right	-
Park21	Right, Close	Left	-

1. Scenario 01

The experiments start with a simple case. The vehicle is straight above the goal position and does not required to change its direction.

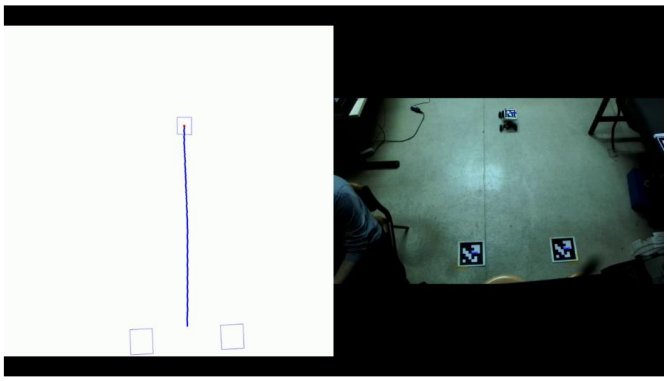

Fig.1: Scenario01-a

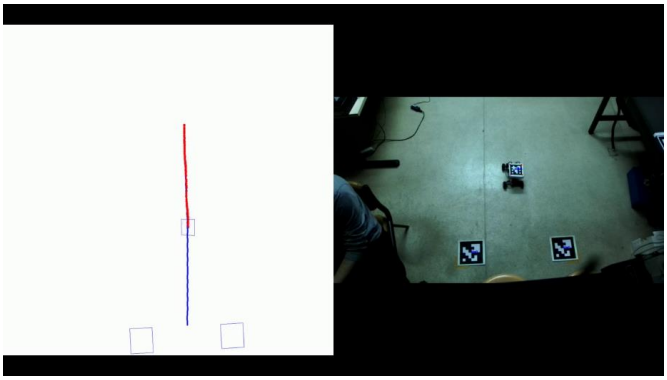

Fig.2: Scenario01-b

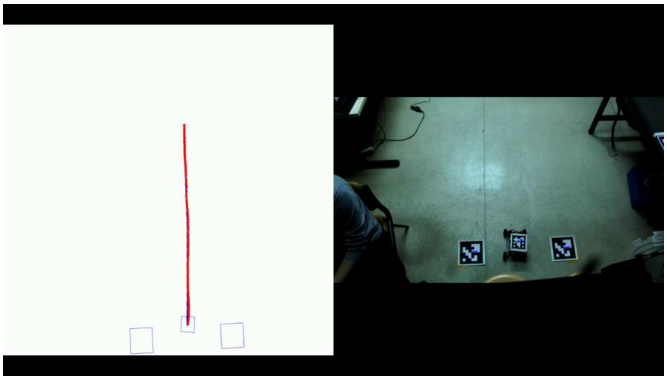

Fig.3: Scenario01-c

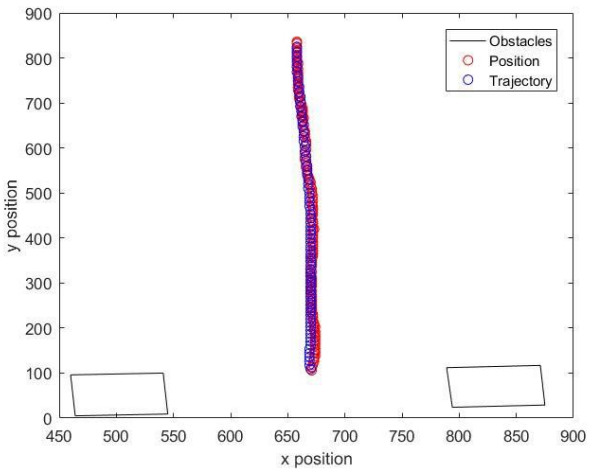

Fig.4: Scenario01-d

2. Scenario 02

In this scenario, the vehicle is on the right side of the experimentation area.

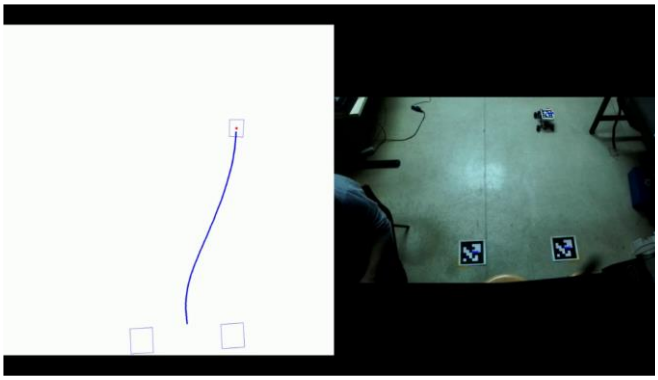

Fig.5: Scenario02-a

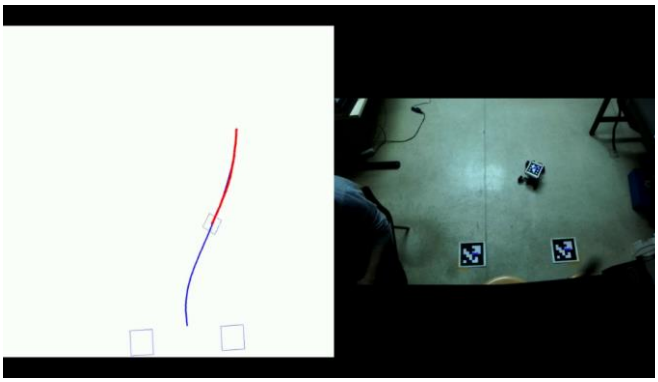

Fig.6: Scenario02-b

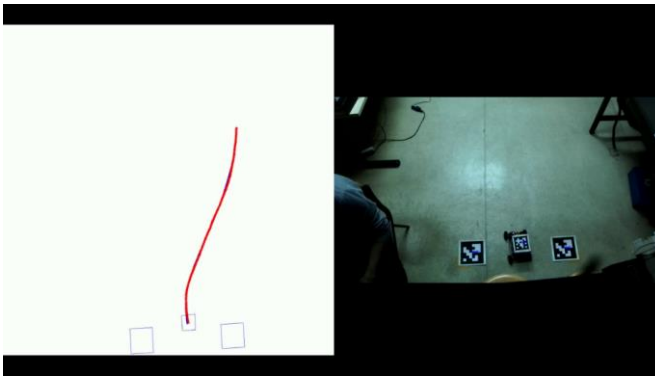

Fig.7: Scenario02-c

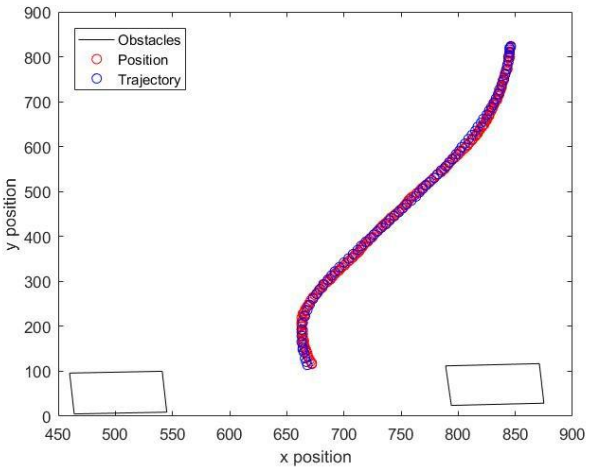

Fig.8: Scenario02-d

3. Scenario 04

In this scenario the starting position and direction of the vehicle is the same with Scenario 01. However, there is an additional obstacle in front of the vehicle. The vehicle re-routes towards the end of its journey.

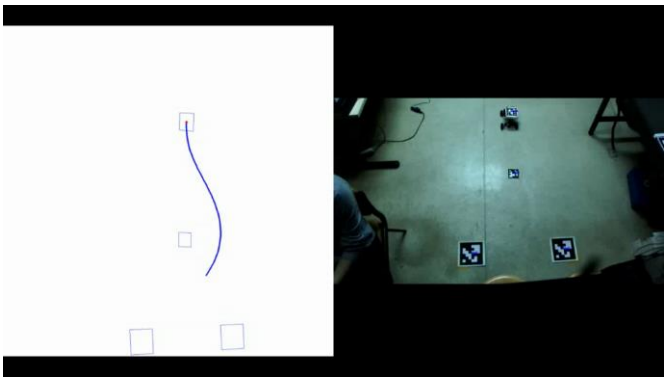

Fig.9: Scenario04-a

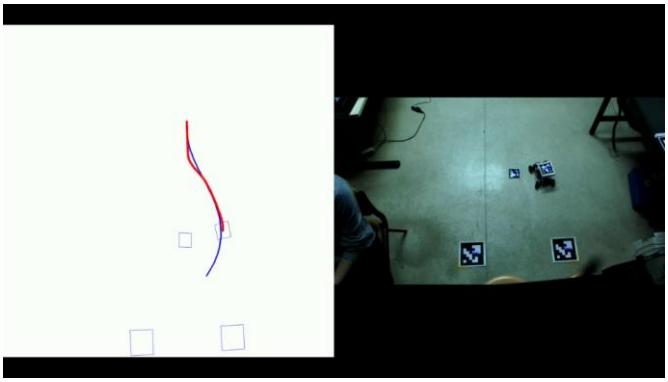

Fig.10: Scenario04-b

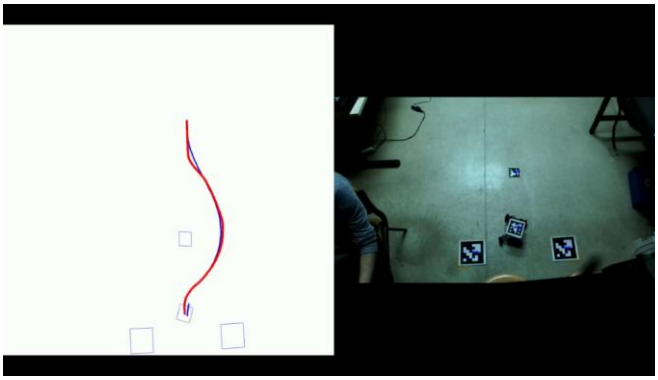

Fig.11: Scenario04-c

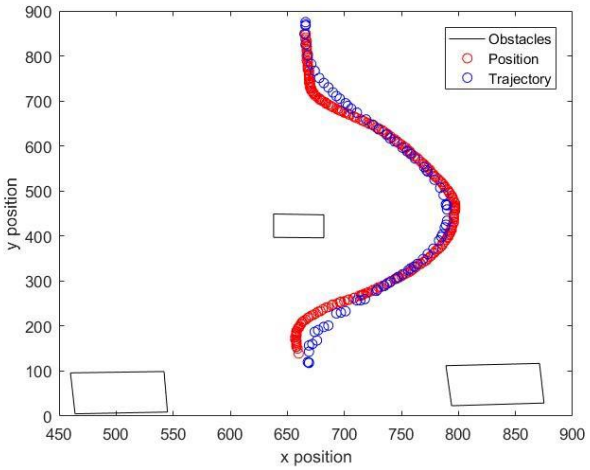

Fig.12: Scenario04-d

4. Scenario 05

This scenario is similar to Scenario04. Only difference is the starting position of the vehicle, which is to right.

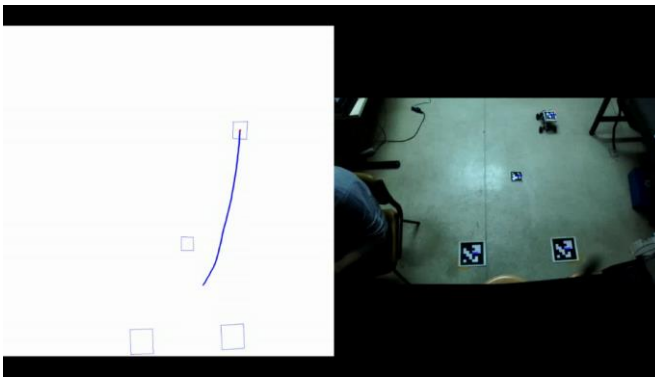

Fig.13: Scenario05-a

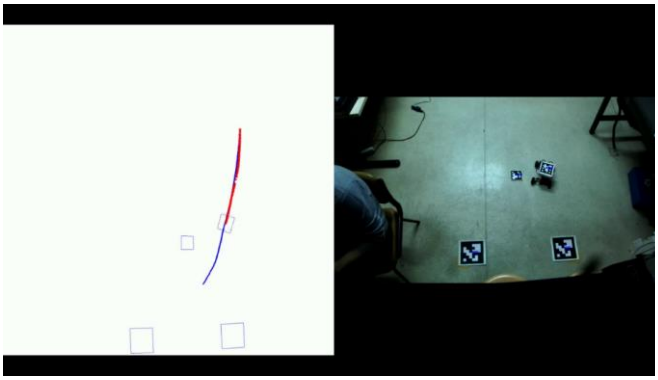

Fig.14: Scenario05-b

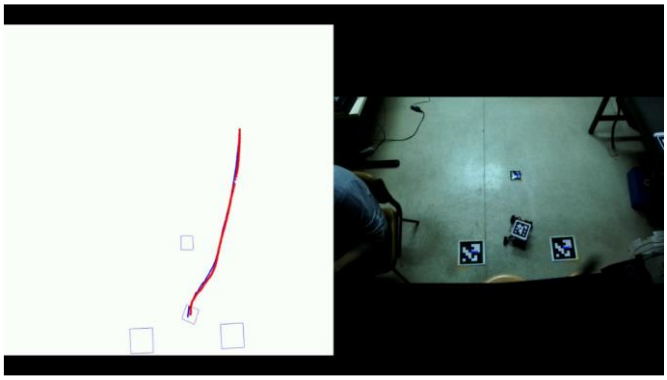

Fig.15: Scenario05-c

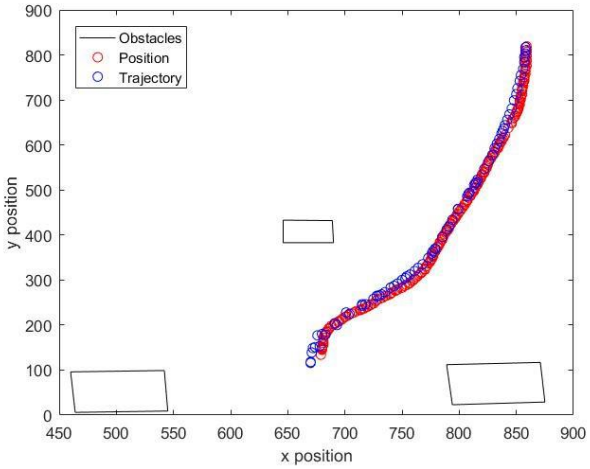

Fig.16: Scenario05-d

5. Scenario 09

This scenario is a version of Scenario01 in which the vehicle's direction is to the left.

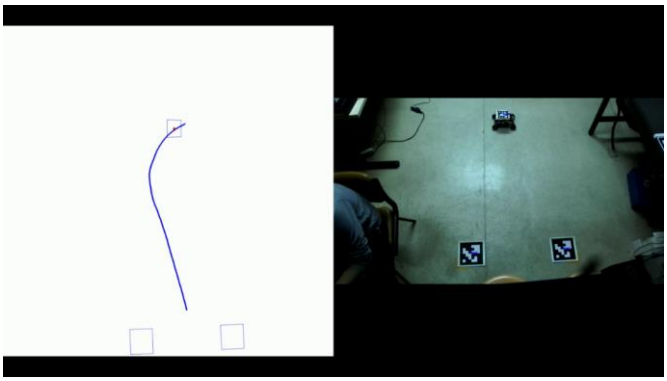

Fig.17: Scenario09-a

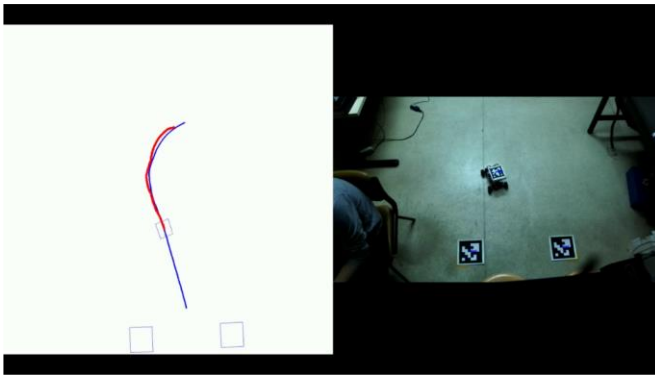

Fig.18: Scenario09-b

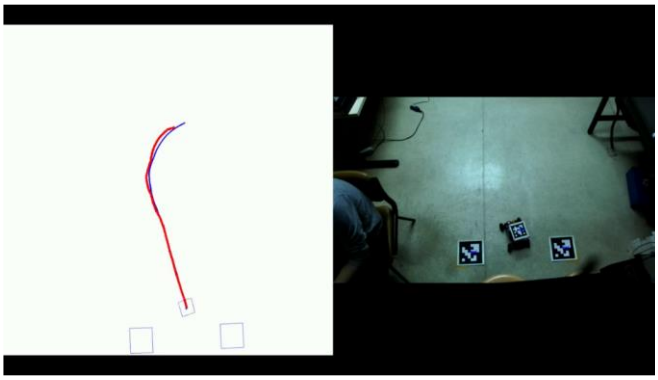

Fig.19: Scenario09-c

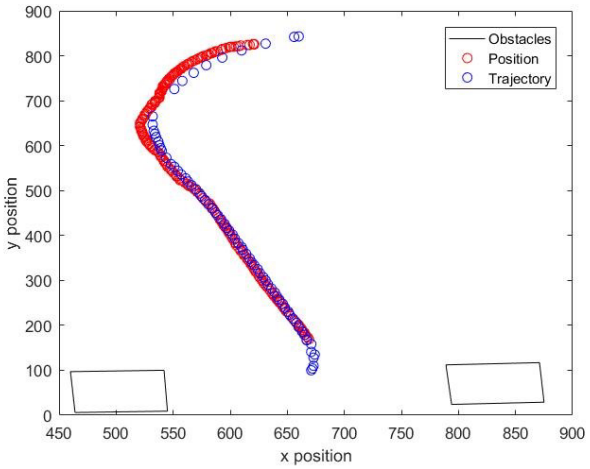

Fig.20: Scenario09-d

6. Scenario 12

In this scenario, the vehicle's starting position is much closer to the destination compared to the previous scenarios, it is on the right side of the experimentation area, facing left. Also, there is an additional obstacle.

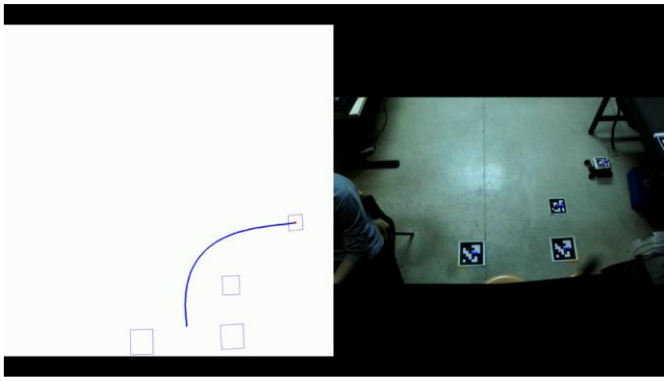

Fig.21: Scenario12-a

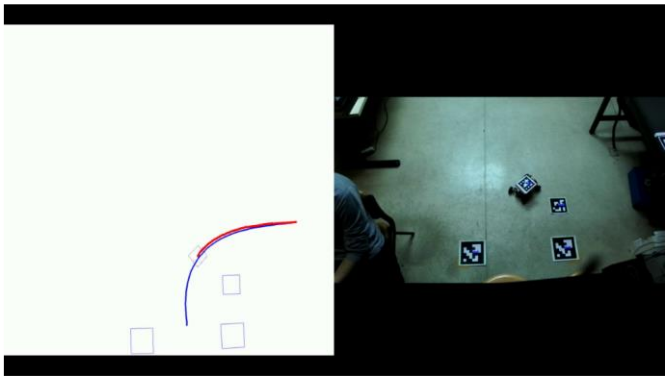

Fig.22: Scenario12-b

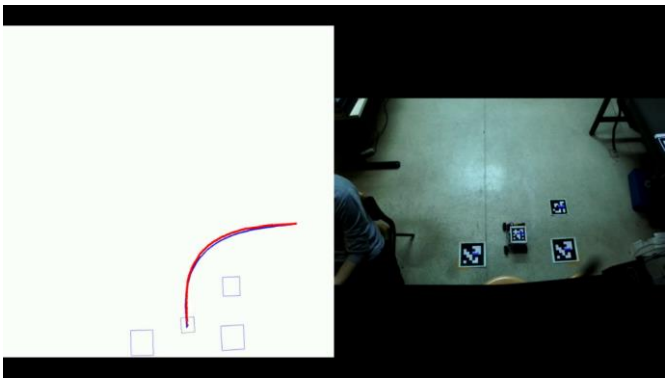

Fig.23: Scenario12-c

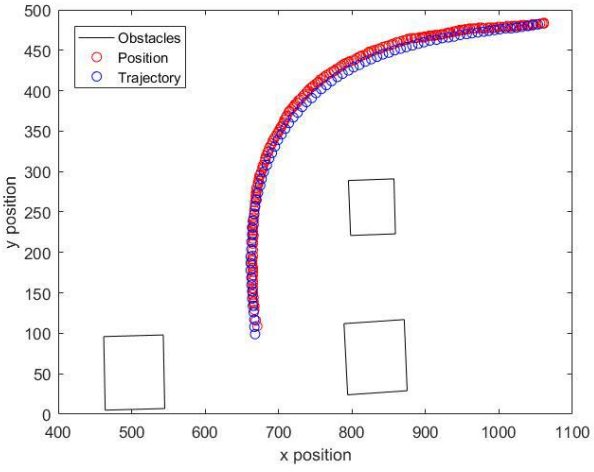

Fig.24: Scenario12-d

7. Scenario 13

In this scenario, the vehicle is one again close to the destination point. It is on the left side of the experimentation area and facing downwards.

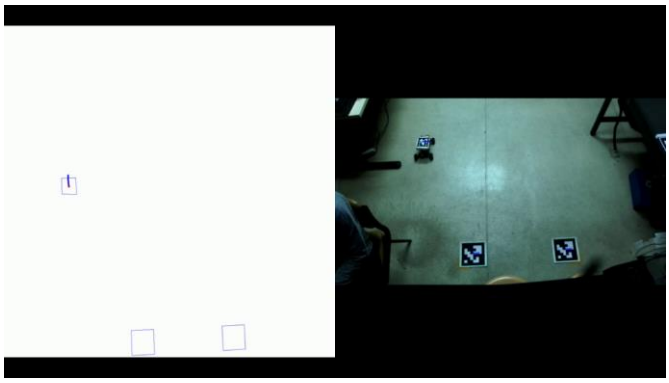

Fig.25: Scenario13-a

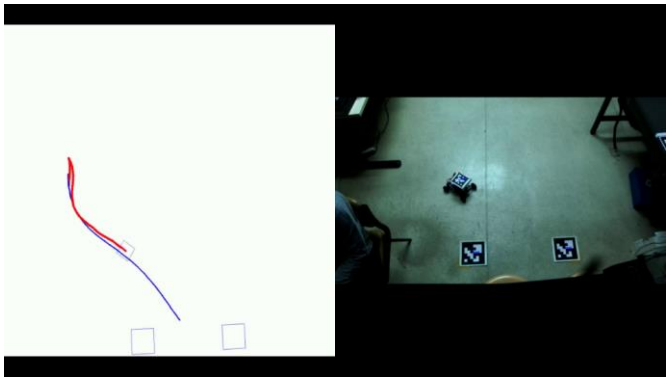

Fig.26: Scenario13-b

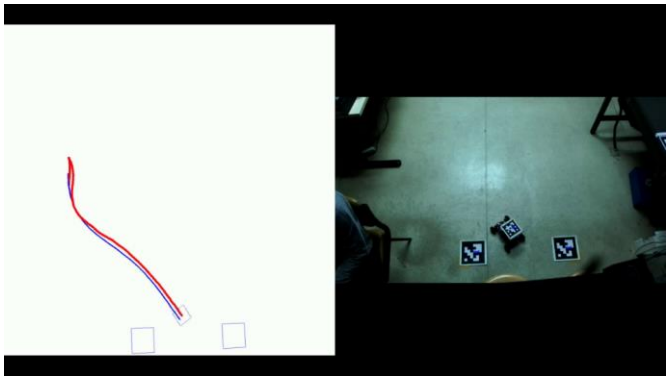

Fig.27: Scenario13-c

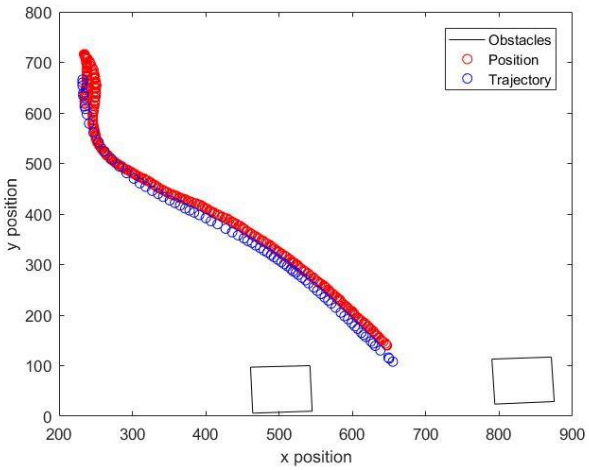

Fig.28: Scenario13-d

IV. Conclusion and Future Work

Our proposed algorithm is a strong alternative to existing algorithms. Its main strengths are working real-time and being realistic. The experimentation was made with a realistic small-scale electric car and engines on our small car is similar to real life electric engines. Hence, our algorithm is scalable and can work at the same efficiency when applied to full-size electric cars.

This study will continue with implementation of well-known path planning algorithms, experimentation on the same setup and in-depth comparison of their results with our algorithm's results.

V. CONCLUSION

The main conclusions of the study should be summarized in a short Conclusions section.

References

[1] U. Ozguner, T. Acarman, and K. Redmill, *Autonomous Ground Vehicles*, Artech House, 2011.

[2] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, 1968

[3] L. E. Kavralu, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, 1996.

[4] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," tech. rep., 1998.

[5] M. Fakoor, A. Kosari, and M. Jafarzadeh, "Revision on fuzzy artificial potential field for humanoid path planning in unknown environment," 2015.

[6] N.D. Richards, M. Sharma, and D. Ward, "Hybrid A*/Automaton Approach to On-Line Path Planning with Obstacle Avoidance," in *AIAA 1st Intelligent Systems Technical Conference*, 2004.

[7] Lego. (2009) Mindstorms 2.0. [Online]. Available: https://www.bricklink.com/v2/catalog/catalogitem.page?S=8547-1#T=S

[8] Lego. (2009) 10287 Intelligent NXT Brick. [Online]. Available: https://lego.fandom.com/wiki/10287_Intelligent_NXT_Brick

[9] OpenCV. (2015) Detection of ArUco markers. [Online]. Available: https://docs.opencv.org/3.1.0/d5/dae/tutorial_aruco_detection.html