



MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

May 2020

Smart Car Parking using Pathfinder

*A Practice School Report submitted to
Manipal Academy of Higher Education
in partial fulfilment of the requirement for the award of the degree of*

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

**SUBMITTED
BY**

STUDENT NAME: Vishal Vimal Kumar
REG. NO: 160905071

Under the Guidance of:

Name of the Internal Guide: Archana Praveen Kumar
Designation: Assistant Professor – Senior Scale
Department: Department of Computer Science



MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

(A constituent unit of MAHE, Manipal)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Manipal

May 31, 2020

CERTIFICATE

This is to certify that the project titled **Smart Car Parking using Pathfinder** is a record of the bonafide work done by Vishal Vimal Kumar (*Reg. No. 160905071*) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B.Tech.) in **COMPUTER SCIENCE & ENGINEERING** of Manipal Institute of Technology, Manipal, Karnataka, (A Constituent Institute of Manipal Academy of Higher Education), during the academic year 2020.

Prof. Archana Praveen Kumar
Assistant Professor – Senior Scale

**Prof. Dr. Ashalatha
Nayak**
*HOD, CSE Dept.
M.I.T, MANIPAL*

ACKNOWLEDGMENTS

Presentation inspiration and motivation have always played a key role in the success of any venture. I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I express my sincere thanks to Dr. D Srikanth Rao, Director of Manipal Institute of Technology. I pay my deep sense of gratitude to Dr. Ashalatha Nayak (HOD) of Department of Computer Science & Engineering, Manipal Institute of Technology to encourage me to the highest peak and to provide me the opportunity to prepare the project.

I am immensely obliged to my friends for their elevating inspiration, encouraging guidance and kind supervision in the completion of my project. I feel to acknowledge my indebtedness and deep sense of gratitude to my guide Mrs. Archana Praveen Kumar whose valuable guidance and kind supervision given to me throughout the course which shaped the present work as its show.

Last, but not the least, my parents are also an important inspiration for me. So, with due regards, I express my gratitude's to them.

ABSTRACT

This paper presents an algorithm for path following so as to supply smooth tracking. Path tracking is that the method concerned with the route to determine speed and steering settings at each instant of time for the vehicle to follow a specific path. A path consists of a bunch of points representing the positional coordinates of a selected route.

The methodology is to implement and evaluate a path tracking algorithm (A star search algorithm) which is frequently utilized in computer science thanks to its completeness, optimality, and optimal efficiency. A* is an informed search algorithm that starts from a particular starting node of a graph, it aims to seek out a path to the given goal node having the littlest cost. It carry out this by maintaining a tree of paths originating at the beginning node and lengthening those paths one edge at a time until its termination criterion is satisfied.

The algorithm described to this point gives us only the length of the shortest path. to seek out the particular sequence of steps, the algorithm are often easily revised in order that each node on the trail keeps track of its former. After this algorithm is implemented, the ending node will point to its former, and so on, until some node's former is that the start node.

This search algorithm can be used in pathfinding. The implementations show the path covered and the actual path taken from start node to end node. These data can be stored and used for a optimal path to be taken in a given circumstances.

Table of Contents			
			Page No
Acknowledgement			i
Abstract			ii
List of Tables			vii
List of Figures			viii
Chapter 1	INTRODUCTION		1
1.1	General introduction to the topic		1
1.2	Motivation		3
1.3	Objective		6
1.4	Target Specification		6
1.5	Project Schedule		7
Chapter 2	BACKGROUND THEORY and/or LITERATURE REVIEW		8
2.1	Problem definition		8
2.2	Introduction to path tracking		9
2.3	Literature Review		10
2.4	Summarized outcome of the literature review		12
2.5	Theoretical Discussion		13
Chapter 3	METHODOLOGY		15
3.1	Introduction		15
3.2	Methodology		17
3.3	Tools used		23
3.4	Conclusion		24
Chapter 4	RESULT ANALYSIS		25
4.1	Introduction		25
4.2	Result analysis		26
4.3	Conclusion		29
Chapter 5	CONCLUSION AND FUTURE SCOPE		30
5.1	Brief Summary of work		30
5.2	Conclusion		32
5.3	Future scope of work		33
REFERENCES			34

ANNEXURES (OPTIONAL)	
PLAGIARISM REPORT	36
PROJECT DETAILS	37

LIST OF TABLES

Table No	Table Title	Page No
1.1	Car parking comparison in minutes	4
2.1	The International Organization of Motor Vehicle Manufacturers	8
4.1	table depicts the time taken from source to destination in different maps	27

LIST OF FIGURES

Figure No	Figure Title	Page No
Figure 1.1	block diagram of automated guided vehicle	3
Figure 2.2	Path tracking with look ahead and overlapped points	9
Figure 2.3	Main definitions of steering angle Φ , heading η , and orientation θ .	10
Figure 2.4	Path tracking with the help of control of steering angle	11
Figure 2.5	Follow the past algorithm	13
Figure 2.6	Follow the carrot algorithm	14
Figure 2.7	Pure pursuit algorithm	14
Figure 3.1	Graph explaining A star algorithm	15
Figure 3.2	Map created by SearchMap function	21
Figure 3.3	virtual bird eye for car parking	22
Figure 4.1	figure depicts map for finding the shortest path	26
Figure 4.2	figure shows the path covered	26
Figure 4.3	optimal path from source to destination	27
Figure 4.4	Test case 2 for optimal path	27
Figure 4.5	Test case 3 for optimal path	28
Figure 4.6	Test case 4 for optimal path	28
Figure 4.7	Test case 5 for optimal path	28

CHAPTER 1

INTRODUCTION

1.1 General Introduction to the topic

Traffic jam or Blockage caused by vehicle is a troubling problem on a global scale, and it has been multiplying aggressively. Car parking problem is a prime donor and has been, still a vital problem with growing vehicle size in the lavish segment and confined parking spaces in urban cities. As the global population rapidly grows, without a well-planned, convenience-driven retreat from the car these problems will worsen. The absence of parking lot makes the parking slot architecture increasingly smaller and narrower. Accordingly, parking conditions are becoming complex progressively, and hence the increasingly higher demand of the parking functioning accuracy is raised, escorting great troubles to various drivers. Automatic parking system (APS) can rise parking safety and utilization rate of parking slot, so it has extensive market application prospects We use path tracking method to enhance the parking procedure.

1.1.1 *Area of Computer Science*

1) Artificial Intelligence

Artificial intelligence (AI) [1] refers to the duplication of human intelligence in machines or instruments that are programmed to think and believe like humans and imitate their actions. The term can also be applied to any machine that exhibits traits related to an individual's mind such as learning and problem-solving. The ideal characteristic of AI is its ability to rationalize and take actions that have the most effective chance of achieving a particular goal.

Artificial intelligence is based on the basis that human intelligence can be explained in a way that a machine can effortlessly mimic it and execute tasks, from the simplest to those that are even more complex. The goals of artificial intelligence include reasoning, learning, and perception.

2) Machine Learning

Machine learning (ML) [2] is the study of computer algorithms that improve automatically through experience. It is seen as a discern of AI. Machine learning algorithms form a mathematical model formed on sample data, so-called "training data", intending to make predictions or decisions without being accurately programmed to do so. Machine learning algorithms are practiced in a wide array of applications, such as computer vision and email filtering where it is tough or infeasible to advance conventional algorithms to carry out the needed tasks.

1.1.2 Applications to the Area of Computer Science

- 1) Artificial intelligence in autonomous [3] vehicle: Just like humans, self-driven cars are required to have sensors to grasp the world around them and a brain to gather, processes and choose particular actions based on information collected. Autonomous vehicles are with advanced/assist tool to gather knowledge, including cameras ,long range radar, and LIDAR. Each of the technologies are worn in different capacities and each gather different information. This information is impractical, unless it is processed, and some form of details is taken based on the collected information. This is where AI comes into play and can be compared to human brain. AI has various applications for these automobiles and among them the more immediate ones are as follows:
 - Directing and controlling the car to gas station or recharge station when it is running low on fuel.
 - calibrate the trips directions based on known traffic conditions to find the quickest route.
 - Incorporate speech recognition for advanced communication with passengers.
 - Natural language interfaces and virtual assistance technologies
- 2) Artificial intelligence for robotics: this will allow us to label the challenges in taking care of the aging population and allow much longer independence. It will drastically reduce, may be even bring down traffic accidents and deaths, as well as enable disaster response for dangerous situations.
- 3) Machine learning for Traffic Prediction [4]: If we want to drop by a new place, we take assist of Google Maps, which shows us the exact path with the shortest route and foresee the traffic conditions. It foresees the traffic conditions such as whether traffic is unobstructed ,heavily congested, or slow moving with the help of two ways:
 - Real Time location and position of the vehicle form Google Map app.
 - Average time that has taken on past days at the same time.Each person who is using Google Map is serving this application to make it better. It takes details from the user and sends back to its database to improve the performance.
- 4) Machine learning in Image Recognition: it is one of the most regular applications of machine learning. It is used to recognize objects, places, persons, digital images, etc. The favoured use case of face detection and image recognition is Automatic friend tagging motion:
Instagram provides us a attribute of auto friend tagging recommendation. Whenever we upload a photo with our Instagram friends, then we spontaneously get a tagging suggestion with name, and the technology supporting this is machine learning's face recognition and detection algorithm.

1.2 Motivation for project

1.2.1 Shortcomings in Reference paper

The paper explains on how to enhance a drive less car by using RTOS (Real Time Operating System) and a smartphone. It is motivated to configure the guidance system of a flexible (Automated Guided Vehicle) AGV as shown in figure 1.1. The driver finds very difficult to park their vehicle in a narrow garage, so it helps to park the vehicle using smartphone via Bluetooth with the range of 100 m, ranges between the car and the smartphone and GPS (Global Positioning System) is also used to know the location. This GPS system will help the user to easily identify the car location. "Car Assist" technology is used to monitor the car driving path and the things happening around the car can be viewed in the smartphone via GPS. It supports live time preview to monitor the car parking garage. The users need not to be present inside the car like some previous generation systems. The proposed work is compared with bench work results and yield very less time to monitor and park the vehicles against the existing system.

Some of the shortcoming in this paper is the recorded path taken by the car. The path taken by the car is important to improve the accuracy of smart parking. We use A star search algorithm for path finding.

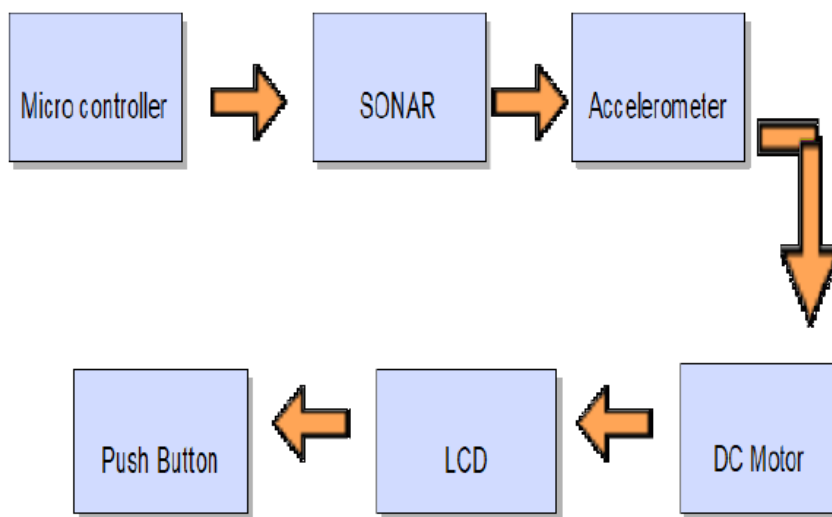


Fig.1.1 Chapter 1 Figure depicts the block diagram of automated guided vehicle

1.2.2 Brief importance of the work in the present context

It is cost efficient. There is no need of any trained person to operate. The circuit is simple. Easily controllable by smart phones even from a distance of 100 m range. Its main advantage is to enhance a drive less car that could be controlled by smartphone and help the user to open the car when the driver reaches near to the car [5]. Remote alert is activated automatically, if the car is parked for a long time in a garage. When compared to other, its time and fuel consuming. Main advantage is if any unauthorized person accessed the car an alarm system is activated and receives a SMS to the smart phone.

Some of the features available in car while driving we could easily stream the music via Bluetooth. The main aim and to have a better solution for smart parking is generated here. Bluetooth is immensely helpful to access the user car from any place and anywhere. In today scenario, parking the vehicles is a challenging task in a narrow parking garage.

The paper also focused on the difficulties faced by the user while parking the car as depicted in table 1.1. It is easy to park the car in a narrow garage. It also has an auto retrieve system that help the car to automatically come out of the parking lot. It has less time consuming and hence it can be installed in all Vehicles. The user could monitor the space required and what happening around the car and be viewed in the smartphone using smart mirror camera. Compared to other we have additional features we can park the car either via GPS or using Bluetooth.

Table 1.1 Chapter 1 Car parking comparison in minutes

NORMAL CAR PARKING SYSTEM		SMART CAR PARKING SYSTEM	
Total no of cars	Total no of time	Total no of cars	Total no of time
1	10	1	6
2	20	2	12
3	30	3	18
4	40	4	24

1.2.3 Methodology to be adopted

The A* search algorithm is an expansion of Dijkstra's algorithm [6] useful for finding the lowest-cost path between two nodes or vertices of a graph. The path may traverse few or any number of nodes attached by edges (arcs) with each edge having an linked cost. The algorithm uses a heuristic, which links an estimate of the minimum cost path from this node to the target node, such that this estimation is never greater than the true cost.

The algorithm shouldn't suppose that all edge costs are the same. It should be possible to start and end on any node, including ones identified as a barricade in the task.

1.2.4 Significance of the methodology

A* is often or usually used for the familiar pathfinding problem in applications such as simulation games but was originally sketched as a general graph traversal algorithm [7]. It finds applications in various problems. Other cases involve an Informational search with online learning. In this paper we will discuss on how A star search algorithm will be used to make a pathfinder.

Consider a example [8] in which there is a square grid or lattice having many obstacles and we are stated a initial cell and a goal cell. We want to reach the goal cell (if possible) from the initial cell as swiftly as possible to achieve it. Here A* Search Algorithm is used to achieve this.

What A* Search Algorithm carried out at each step it chooses the node according to a value- 'f' that is a parameter equal to the sum of two parameters as mentioned- 'g' and 'h'. At each step it chooses the node/cell having the lowest 'f', and processes that node or a cell.

We define 'g' and 'h' as directly as possible as shown:

g = the action or movement cost to move from the beginning point to a given point or square on the grid or lattice, following the path or trail made to get there.

h = the estimated action or movement cost to move from that given square on the grid or lattice to the final destination. This is often assigned to as the heuristic. We really don't know the correct or actual distance until we find the path or trail, because all sorts of obstacles can be in the way.

1.3 Objective

1.3.1 Primary Objective

To use A star search algorithm to find the shortest path between nodes or graphs and use this as a pathfinder. It is an informed search algorithm, as it uses information about path cost and also uses heuristics to find the solution.

1.3.1 Secondary Objective

To enhance a drive less car by using path tracking algorithm. A human driver or operator drives the given path initially, while a computer is constantly recording the position or location , velocity or speed, orientation or direction, and steering angle. These details are basically used to command the vehicle whenever it autonomously travels along the given trail. On condition that the vehicle gets astray, for instance, as a result of keeping away from an obstacle or due to noise produced within the positioning sensors. We find the simplest possible actions or path it should absorb a selected situation with the aid of this algorithm.

1.4 Target Specification

This technology aids the drivers in parking their vehicles, making it effortless for drivers who find parking difficult. It automatically handle the steering, leaving the driver at liberty to concentrate on handling the brakes and accelerator and checking the environment. The system make use of footage from the surround View Monitor, which offers drivers a virtual bird's-eye perspective of the vehicle, permitting drivers to park their vehicles more effortlessly.

The driver examines the virtual bird's-eye view of the vehicle on the surround View Monitor and regulates the position for the car to park. The vehicle then automatically manoeuvre itself into position. The driver follows the instructions and audio signal shown on the screen and handles the brakes and accelerator. By moving onward or reversing while regulating the speed, the car can be parked in the given area.

1.5 Project Schedule

- *January 2020*
 - Preparing synopsis.
 - Understanding more about IoT.
- *February 2020*
 - Learn basics and understanding of Microcontroller.
 - Understanding global positioning system.
 - To learn the benefits of Bluetooth and android usage.
 - Tutorials on Reinforcement learning.
- *March 2020*
 - To learn about Unsupervised Machine Learning.
 - Effects of Reinforcement learning.
 - Learning about A star search algorithm.
 - Learn about follow the past algorithm.
- *April 2020*
 - Using A star search algorithm for pathfinding
 - Submission of report & evaluation (4 Months project).
 - Writing and publishing a paper.

CHAPTER 2

BACKGROUND THEORY / LITERATURE REVIEW

2.1 Problem definition

India was the fourth largest car maker or manufacturer in the world in 2018 [9]. Indian auto makers or manufacturers made a record 30.9 million motor vehicles in 2018-19 (Apr-Mar) including 4.03 million passenger vehicles. India was the largest manufacturer or maker of three-wheelers (1.27 million units) and also the seventh biggest commercial vehicle (1.11 million units) maker in 2018-19 while India's two-wheeler makers or manufacturers rolled out 24.5 million units during the same fiscal. Construction vehicle production was approximately 59 000 units in 2014. 2.17 million passenger vehicles were sold in India in 2017-18. Figure 2.1 shows the passenger vehicle sales between 2016 to 2017.

Due to the proliferation within the number of vehicles on the road, traffic problems are sure to exist. this is often thanks to that the present transportation infrastructure and parking lot facility developed are unable to deal with the influx of vehicles on the road.

On day to day, it is evaluated that 70% of vehicles on the road in urban area of major cities have bare minimum parking spot. This causes not only waste of our time and fuel for drivers while trying to find the parking. parking lot is expensive to make or build and sometimes Scarce resource. To alleviate the current problems, the smart parking system has been developed.

Table.2.1 Chapter 2 OICA (The International Organization of Motor Vehicle Manufacturers

Passenger Vehicle Sales (2016-17)

Type	Sales (thousands)
Passenger car	2103
SUV	762
Van	182

2.2 Introduction to Path Tracking

Path tracking is that the method bothered with the way to find the speed and steering settings at each instant of time so that the vehicle can follow a particular path. A path contains of a collection of points showing the positional coordinates of a specific path as shown in figure 4.1. Often when executing a path tracking algorithm, one also needs to execute a path recording unit liable for saving all the position that constitutes the path. an individual's operator then has the likelihood to manually direct the robot along some track while the path recording unit stores the knowledge about the path. the path tracking algorithm also requires to manage unplanned positional or direction changes from the path. Such changes are often caused by odometrical mistakes of some kind, or by new bumps occurring on the path that has to be avoided.

This implementation is quite simple and straightforward to execute. the complete method of path tracking, and path planning has been strongly implemented on Nomad 200 mobile robot. The reference path is clustered into a collection of equispaced points. Using this latest path, the path tracking algorithm assess the steering commands for the robot implementing the pure pursuit algorithm [10].

There are many other types of path tracking algorithms accessible these days. These algorithms can be implemented in smart parking system to improve the parking time and steering angle of the given vehicle.

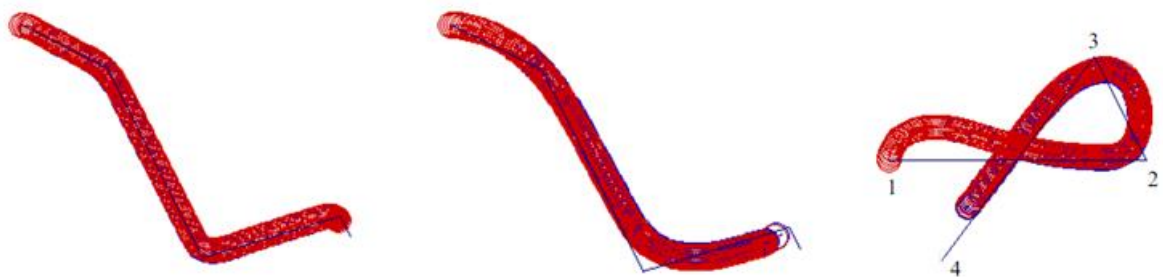


Fig.2.2 Chapter 2, Path tracking with look ahead and overlapped points

2.3 Literature review

2.3.1 Recent Developments in the work area

Follow the past algorithm [11] was used for smart parking. The main focus of the algorithm are direction its heading, orientation and steering angle. The heading η (eta) is determined to be the direction of the front a part of the vehicle. The steering angle Φ (phi) can be determined as the angle between the nose and rear sections of the vehicle. The orientation θ (theta) is determined as the direction within which the vehicle would advance if the steering angle was zero and therefore the baseline was maintained, as illustrated by the dashed vehicle in Figure 2.3.

θ (theta) are often calculated as difference between the heading (or nose) of the vehicle and half the steering perspective of the vehicle, i.e. $\theta = \eta - (\Phi / 2)$. The heading η (eta) and also the orientation θ (theta) are expressed during this given global coordinate system.

Firstly there will be a manual driving along the trail, the steering angle and orientation are documented alongside the position at every turn. The recorded orientation θ' (theta dash) and therefore the recorded steering angle Φ' (phi dash) are engaged by the Follow the Past method, which consists of three independent behaviour's:

ϕ_β : Turn towards the recorded orientation θ' . (2.1)

ϕ_γ : Mimic the recorded steering angle ϕ' . (2.2)

ϕ_α : Move towards the path. (2.3)

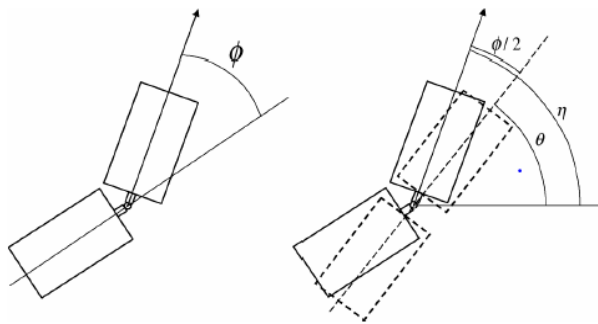


Fig.2.3 Chapter 2, Main definitions of heading η (eta) , and orientation θ (theta) and steering angle Φ (phi)

2.3.2 Brief Background Theory

The given vehicle initially operates on the present input values ; estimated shortest distance to the trail ,orientation and steering angle. ϕ_α uses recorded or documented closest position (x' , y') and true position (x , y) as input data. ϕ_β uses and actual orientation θ and recorded orientation θ' as input data. ϕ_γ uses the recorded or documented steering angle Φ' as input data. The given three behaviours are combined into one movement as shown in Figure 2.4. The three individualistic behaviours ϕ_α , ϕ_β and ϕ_γ operate as shown below.

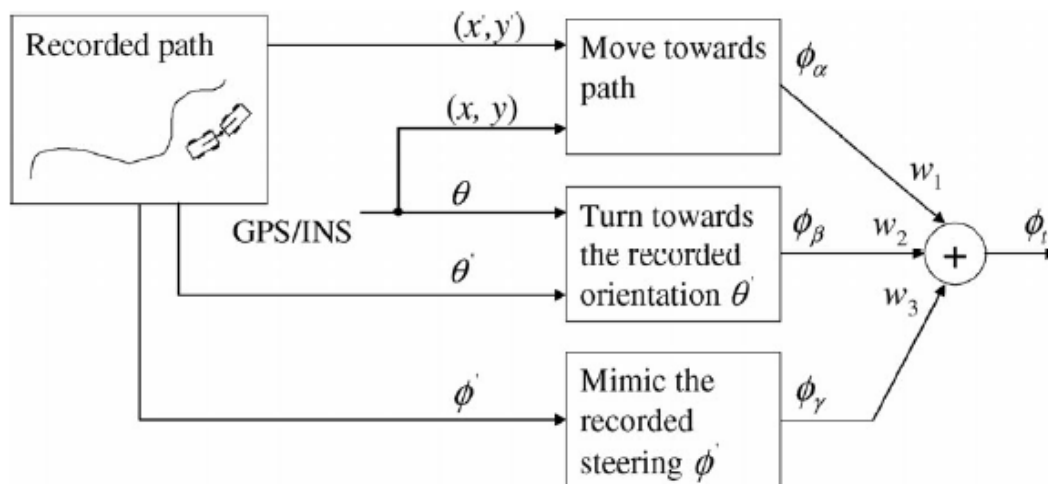


Fig.2.4 Chapter 2, Path tracking with responsive control of steering angle ϕ_t

2.4 Summarized outcome of the literature review

2.4.1 ϕ_β : *Turn towards the recorded orientation*

The angle θ' (theta dash) is explained as the recorded orientation at the near or closest point on the recorded path. this point is named the path point. ϕ_β is computed because of the difference between the current orientation θ and therefore the recorded orientation θ' .

$$\phi_\beta = \theta' - \theta \quad (2.4)$$

2.4.2 ϕ_γ : *Imitate the recorded steering angle*

Basically the given behaviour returns the recorded steering angle Φ' at the given time point.

$$\phi_\gamma = \Phi' \quad (2.5)$$

By utilizing the given recorded steering angle, the curvature of the path is automatically included within the final steering command. this is often an excellent advantage compared with methods like Pure Pursuit algorithm [12] and Follow the Carrot algorithm[13].

2.4.3 ϕ_α : *Move towards the given trail*

This behaviour is accountable for bringing the vehicle back to the path if the vehicle deviates from the trail for some reason. Such a deviation are often caused by noise within the position signal or by the obstacle-avoidance system, which may be a separate behaviour, not described in any detail in this report. are often implemented in many ways.

2.5 Theoretical discussion

A considerable amount of algorithms for path tracking are narrated in the automation articles. Traditional algorithms, like Follow the Carrot, use environment information to evaluate steering instruction that prepare a vehicle to follow a pre-defined path relatively. These algorithms are well known or established to cut corners, since they do not notably take into account the authentic curvature of the path. In this chapter we employ a novel algorithm that makes use of recorded steering commands to beat this situation.. The algorithm is implemented both on a simulator for autonomous forest machines and a physical small-scale robot

2.6 Conclusion

The three behaviours ϕ_α , ϕ_β and ϕ_γ return a recommended steering angle, pointing towards to fulfilling the goals of the respective or individual behaviours. These three values are fused into one value ϕ_t .

By addition of equation 2.1, 2.2 and 2.3. we get

$$\phi_t = \phi_\beta + \phi_\gamma + \phi_\alpha \quad (2.6)$$

The advanced algorithm has been examined on a given pioneer robot and was compared to the already existing implementations of Pure pursuit method and follow the carrot method . The Pioneer robot is equipped with a RTKDGPS for positioning and a magnetic compass combined with a gyro to measure the heading of the vehicle.

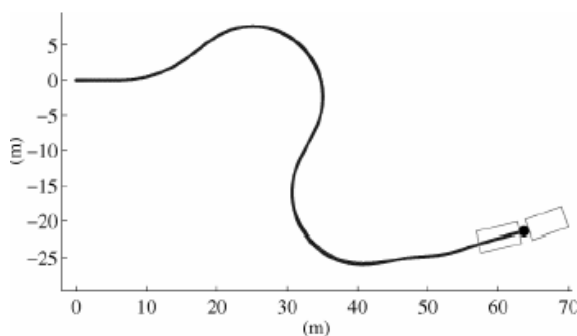


Figure 2.5 Chapter 2, Follow the past

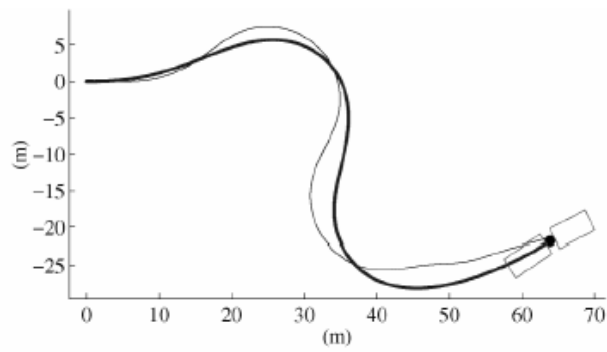


Figure 2.6 Chapter 2, Follow the carrot

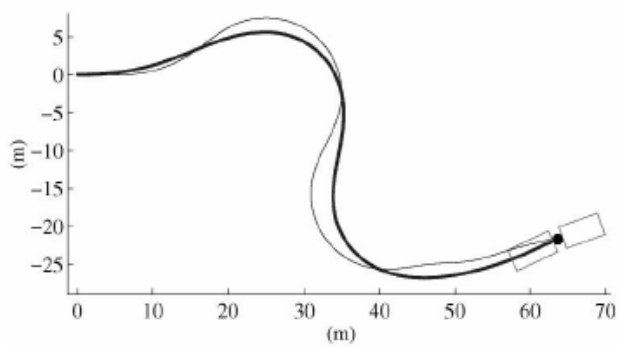


Figure 2.7 Chapter 2, Pure pursuit

The thick line is the vehicle capable of perfectly following a recorded path (thin line) implemented using follow the past algorithm as shown in the given figure 2.5, there is not much of deviation from the given path.

Figure 2.6 and 2.7 shows the behavior of vehicle when using follow the carrot and pure pursuit algorithm under same conditions. This path has some sharp turns, which makes it difficult for both Follow the Carrot and Pure Pursuit, as they tend to chop corners rather than following a highly curved path. Under ideal conditions, this problem is avoided by the Follow the Past algorithm.

CHAPTER 3 METHODOLOGY

3.1 Introduction

In this chapter we will discuss about execution of A star search algorithm. A star algorithm, finds the most optimal path [14] that it can take from the source or starting point in reaching the final point or destination. It realizes which is the best trail that can be taken from its actual or current state and reach its final point or destination.

The given algorithm has 3 variables which are explained below:

- 1) F – F is the variable of A star which is the sum of the other variables H and G and is the minimum or least cost from one node to the succeeding or next node. This variable is in charge for guiding us to find the most optimal path from our starting point or source to final point or destination.
- 2) H – H is the given variable for heuristic/estimated path between the actual or current code to the final point or destination node. This cost is not genuine but is, in reality, a estimate cost will be used to find the most optimal path between our source or starting point and destination or final point.
- 3) G – G is the given variable that depicts cost of node from one node to the other subsequent node. This variable changes for every node as we move towards upward direction to find the most optimal path.

So $F = G + H$. consider a given example which helps us to understand this algorithm in a detailed manner. Suppose we have a small graph as shown in figure 3.1 with the vertices: A, B, E, S where S is the source or starting point and E is the destination or final point.

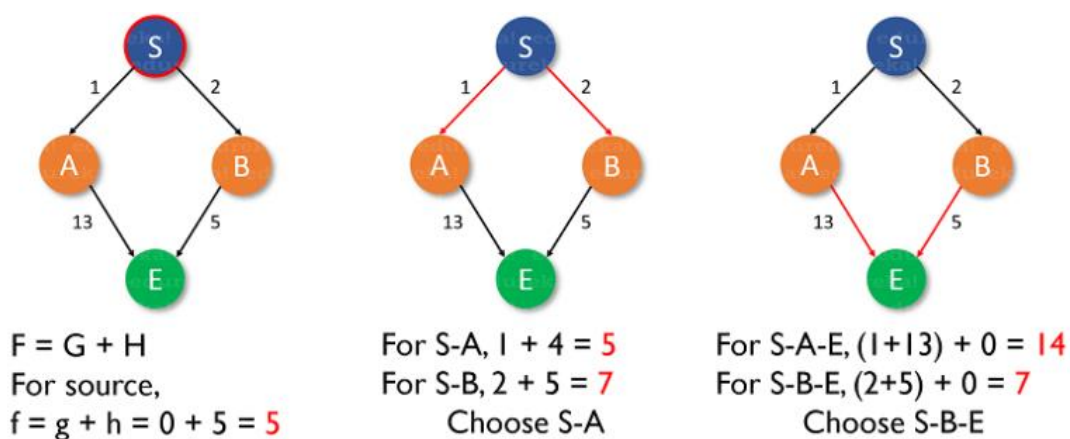


Figure 3.1, Chapter 3 Graph explaining A star algorithm

NOTE: cost to enter the initial point or source and final point or destination is always zero.

The heuristic values are shown below:

- 1) S \rightarrow 5
- 2) A \rightarrow 4
- 3) B \rightarrow 5
- 4) E \rightarrow 0

Using the given the formula and calculate the displacement from the initial point or source to the final point or destination now. $F = g + h$ where g is the variable for cost to travel and the variable h is the given heuristic value.

To reach initial point or Source:

$$F(S) = [0 + 5] = 5$$

The trail from S to other vertices(A and B):

$$f(S-A) = [1 + 4] = 5$$

$$f(S-B) = [2 + 5] = 7$$

So, first and foremost we select the path of S \rightarrow A because it is the minimum.

The trail from A and B to the final point or Destination:

$$f(S-A-E) = [1 + 13] + 0 = 14$$

$$f(S-B-E) = [2 + 5] + 0 = 7$$

After estimation, it is found that B later has given us the minimum or least path. So, we adjust our least or minimum trail to S-B-E and have arrived at our destination. This is how we implement the formula to find out the most optimal path.

3.2 Methodology

A* is often or usually used for the familiar pathfinding problem in applications such as simulation games but was originally sketched as a general graph traversal algorithm. In introduction we explained A star algorithm with a given example using a formula. The algorithm can be explained below. Firstly, we need to create 2 lists that will help us understand the path. The list can be named as openSet and closedSet list.

A* search algorithm():

- 1) Add start (or initial) node to list. (A star finds a path from start to goal)
- 2) For all the nearest nodes, find the least or minimum cost F node
- 3) Switch to the closedSet list
 - For eight nodes alongside to the present node
 - If the node is inaccessible, ignore it. Else
 - 1) If the node is not found on the openSet list, move it to the openSet list and calculate f, g, h.
 - 2) If the node is on the openSet list, check if the trail it offers is less than the actual or current path and change to it if it does so.
- 4) Stop working when
 - You find the destination
 - You may not find the destination or final point after going through all possible points.

3.2.1 Detailed Methodology

The algorithm can be explained in a detailed way with the help of pseudocode as shown below.

```
function AStar_algo(originateFrom, current)
    total_route := {current}
    while current in originateFrom.Keys:
        current := originateFrom[current]
        total_path.prepend(current)
    return total_route

// A* finds a path from source or start to final destination or goal.
// h is the heuristic function. h(n) approximates the cost to reach final destination or goal from
// node n.
function A_Star(starting_point, goal, h)
    // The set of found nodes that may need to be (re-)expanded.
    // At first, only the starting node is known.
    // This is usually carried out as a min-heap or priority queue rather than a hash-set.
    openSet := {starting_point}

    // For node n, originateFrom[n] is the node immediately preceding it on the cheapest path
    // from start
    // to n currently known.
    originateFrom := an blank map

    // For node n, CostScore[n] is the cost of the economical path from start to n currently
    // known.
    CostScore := map with default value of Infinity
    CostScore[starting_point] := 0

    // For node n, short_path[n] := CostScore[n] + h(n). short_path[n] depicts our present best
    // guess as to
    // how shorten a path from initial or start to destination or finish can be if it goes through n.
    short_path := map with default value of Infinity
    short_path[start] := h(starting_point)
```

```

while openSet is not blank
    // This functioning can happen in O(1) time if openSet is a priority queue or min-heap
    current := the node withn openSet possess the lowest short_path[] value
    if current = goal
        return AStar_algo(originateFrom, current)

    openSet.Remove(current)
    for each neighbour of current
        // d(current,neighbour) is the density of the edge from current to neighbour
        // tentative_CostScore is the length from start or initial point to the neighbour through
current
        tentative_CostScore := CostScore[current] + d(current, neighbour)
        if tentative_CostScore < CostScore[neighbour]
            // This path to neighbour is better than any previous one. Record it!
            originateFrom[neighbour] := current
            CostScore[neighbour] := tentative_CostScore
            short_path[neighbour] := CostScore[neighbour] + h(neighbour)
            if neighbour not in openSet
                openSet.add(neighbour)

// Open set is blank but goal was never unattainable

return failure

```

Remark: In the given pseudocode, if a node is attained by one path, withdrawn from openSet, and afterwards attained by a cheaper path, it will be put on to openSet once more. This is necessary to assure that the path returned is ideal if the heuristic function is relevant but not rational. If the heuristic is rational, when a node is withdrawn from openSet the trail to it is guaranteed to be ideal so the test 'tentative_CostScore < CostScore[neighbour]' will always fail if the node is reached again.

3.2.2 Design and modelling the obstacles

A pathfinding method inspects a graph by starting at one vertex and searches neighbouring nodes until the destination or final node is attained, generally with the purpose of discovering the economical route. Although graph searching methods such as a (BFS) breadth-first search would discover a route if given adequate time, other methods, which "scout" the graph, would tend to reach the destination or final point sooner. For the A star algorithm mentioned in section 3.2.2 we create obstacles to test the working of it.

We create a map, the pseudo code explains about how the map is created

```
function SearchMap(cols, rows, x, y, w, h, permitdiagonals, wallRatio ) {  
  
    this.cols = cols;  
    this.rows = rows;  
    this.grid = [];  
    this.path = [];  
  
    this.x = x;  
    this.y = y;  
    this.w = w;  
    this.h = h;  
  
    // Making a 2D array  
    for (var i = 0; i < cols; i++) {  
        this.grid[i] = [];  
    }  
  
    for (var i = 0; i < cols; i++) {  
        for (var j = 0; j < rows; j++) {  
            var isWall = random(1.0) < wallRatio;  
            this.grid[i][j] = new Spot(i, j, x + i * w / cols, y + j * h / rows, w / cols, h / rows,  
isWall, this.grid);  
        }  
    }  
  
}
```

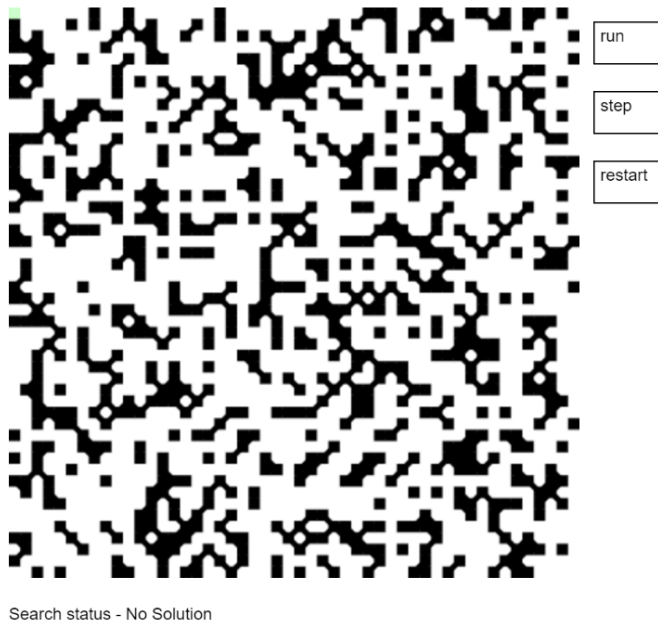


Figure 3.2, Chapter 3 Map created by SearchMap function

The given map shown in Figure 3.2 consist of obstacles which are black in colour. The green colour shown in upper left corner is the starting node. It searches for the path by avoiding the black coloured obstacles in order to reach the destination.

A* uses this heuristic to enhance on the behavior in relation to Dijkstra's algorithm. When the heuristic checks to 0, A* is similar to Dijkstra's algorithm. As the heuristic measure increases and coming up to the actual distance, A* proceeds to find optimal paths, but executes faster (an account of examining fewer nodes). When the cost of the heuristic is exactly the actual distance, A* evaluates the fewest nodes. (However, it is generally illogical to address a heuristic function that always determines the actual distance, as the same correlation result can often be attained using effortless calculations – for eg: using Manhattan distance on top of Euclidean distance in 2D space.) As the rate of the heuristic increases, A star examines fewer nodes but no longer assure a best path. Applications (such as video games) this is adequate and even enticing, in order to keep the algorithm running quickly.

3.2.3 Assumptions made

We need to establish a path tracking algorithm for autonomous vehicle parking. Initially the human drives parks the vehicle with a given path once. We record the position , orientation, velocity, and steering angle. These details are used to control the vehicle each time it autonomously travels along the path during parking procedure. The A star algorithm has proven to be much more appropriate than other traditional algorithms for path tracking.

The system is essentially configured in the same way as the (AVW) Around View Monitor. It consist of four cameras on the car body (the rear, front, and sideview mirrors). The system transforms video footage from the cameras and compounds it to a virtual bird's-eye view image as shown in figure 3.3.

The system checks the conditions of the environment by the virtual bird's-eye view image as shown in figure 3.3 and helps by guiding the driver along a route to the designated parking position. So, each time the driver parks the vehicle the path will be recorded. After finding subsequent path taken, we use the best and optimal path taken in the given scenario.

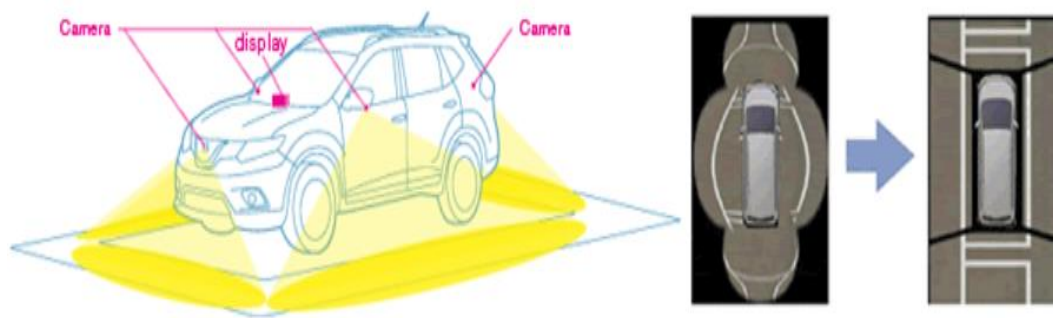


Figure 3.3, Chapter 3 virtual bird eye for car parking

3.3 Tools used

In order to implement A star search algorithm, we used JavaScript, JavaScript is one of the effortless, flexible and compelling languages used to broaden functionality in websites. JS Development Services helps in processing and on-screen visual effects and calculating data on web pages. The programming language also assists in extended functionality to websites using third party scripts among several other handy features.

Some of the advantages of JavaScript are:

- 1) JavaScript is comparatively fast for the end user- on the client side the code gets executed, processing and results is completed almost immediate depending on the task (tasks in JavaScript on web pages are usually easier so as to avoid being a memory hog) as it doesn't need to be prepared in the site's web server and returned to the user consuming local as well as server bandwidth.
- 2) JavaScript is a client-side language- On the user's processor JavaScript code is executed instead of the web server thus it recovers bandwidth and weight on the web server.
- 3) JavaScript is an easy language to learn- The JavaScript language is easy and simple to learn and provides syntax that is nearly English. It uses the DOM model that brings plenty of predefined functionalities to the different objects on pages making it a effortless to develop a script to solve a custom purpose.
- 4) Extended functionality to web pages- third party add-ons allow JavaScript developers to write snippets of JavaScript that can be executed on particular web pages to extend its functionality. If we use a website that requires a certain feature to be admitted, we can write it by ourselves and use an add-on to implement it on the web page.
- 5) Easy to debug and test- understanding of syntax in JavaScript is easy. Any person can learn it effortlessly and use it to develop dynamic and scalable websites.
- 6) Platform independent- Any JS-enabled browser can recognize and clarify JavaScript code. Any JS code can be accomplished on different types of hardware a JS program is written for.

3.4 Conclusion

The A star search algorithm is widely used for path finding and also graph traversal. The given algorithm efficiently plots a optimal path between multiple nodes or points on the given graph. But A star is slow and also takes a lot of space as it saves all the available paths that are feasible to us. This makes other quicker algorithms have an upper hand over A star but it is nevertheless, one of the best algorithms out there.

Autonomous vehicles are a rapidly developing field. The application of autonomous vehicles is expected to be widely used in urban areas, industry, and airport terminals, etc. An autonomous vehicle is adequate of sensing its environment, navigating and decision making by itself. Trajectory tracking is an essential aspect of autonomous vehicles. The plan behind trajectory tracking is the capability of the vehicle to go along a predefined path with 0 steady state error. The struggle arises due to the nonlinearity of vehicle dynamics. The use of A star algorithm helps to reduce these errors by giving an optimal path.

CHAPTER 4

RESULT ANALYSIS

4.1 Introduction

In this we will discuss about the optimal path taken from one point or starting from a specific node on the graph to the final point or destination assigned. A star is an well-informed search algorithm, or a BFS, meaning that it is prepared in terms of weighted graphs: beginning from a specific starting node of a graph, it plans to find a path to the given destination node having the least cost (smallest distance travelled, shortest time, etc.). It achieve this by maintaining a tree of paths derived at the initial node and broadening those trails one edge at a time until its expiry criterion is satisfied.

The algorithm expressed so far gives us only the distance of the shortest path. To search the actual array of steps, the algorithm can be simply revised so that each node on the path (or trail) monitor its predecessor. After this algorithm is executed, the ending node or final node will pinpoint to its predecessor, and so on, until they find some node's predecessor is the start node.

4.2 Result analysis

Consider the given map as shown in figure 4.1, in this we need to find the optimal path from starting node to destination and time taken.

Step1: figure 4.1

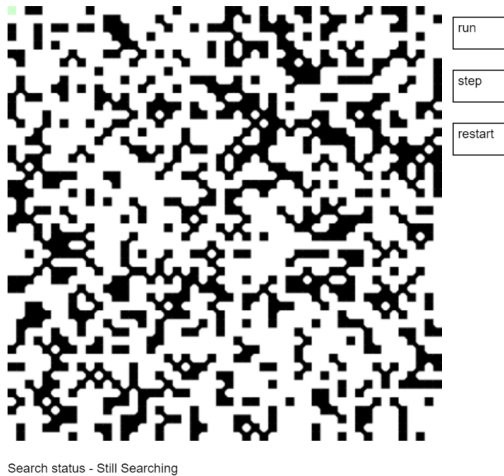


Figure 4.1, Chapter 4, figure depicts map for finding the shortest path

Step2 : In figure 4.2 the red colour represents the path covered and the green colour depicts the approximate path covered .

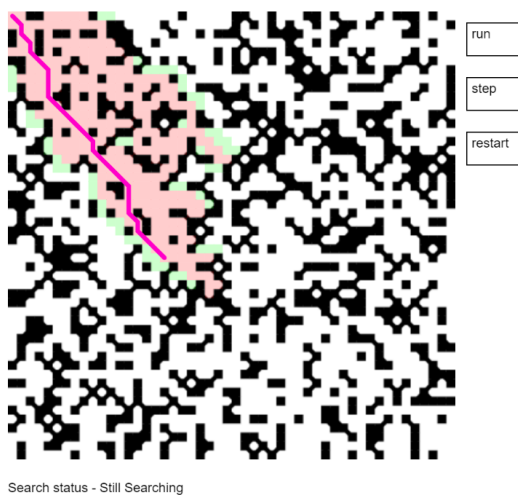


Figure 4.2, Chapter 4, figure shows the path covered

Step 3: In figure 4.3 the purple line shows the shortest path taken from initial point or source to final point or destination.

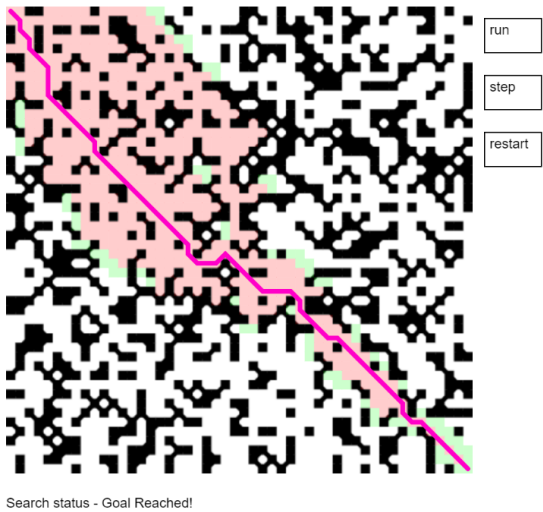
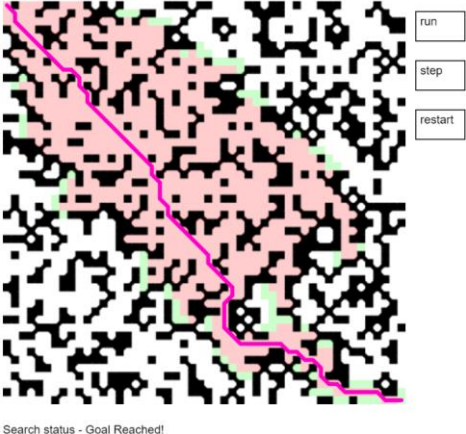


Figure 4.3, Chapter 4, optimal path from source to destination

4.2.1 Tabulated results

Test cases	Time taken
Test case 1- map shown in figure 4.1	9.83 seconds
Test case 2  <p>run</p> <p>step</p> <p>restart</p> <p>Search status - Goal Reached!</p> <p>Figure 4.4, chapter 4, Test case 2 for optimal path</p>	13.49 seconds

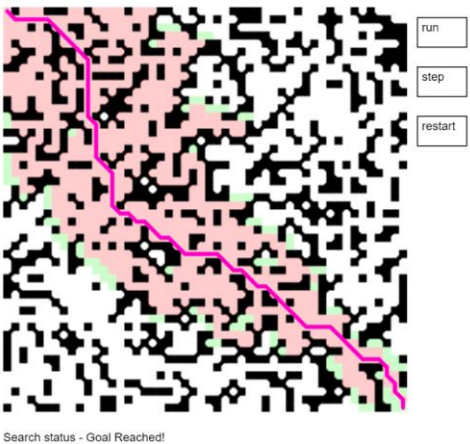
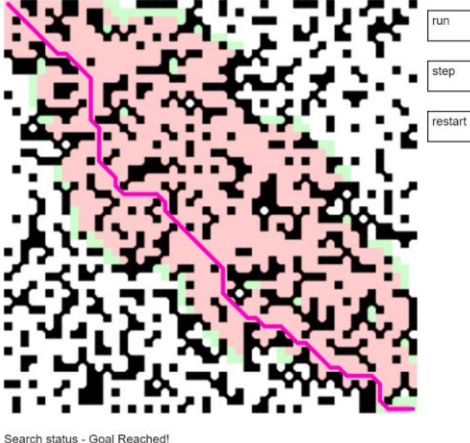
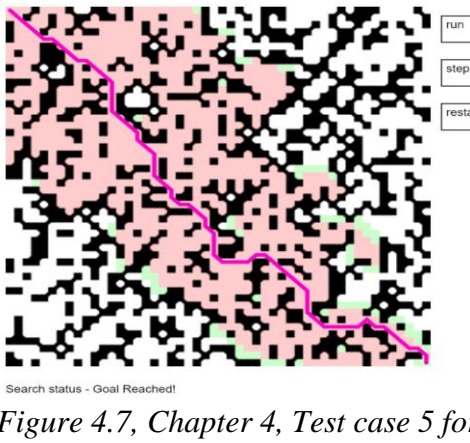
<p>Test case 3</p>  <p>Search status - Goal Reached!</p> <p><i>Figure 4.5, Chapter 4, Test case 3 for optimal path.</i></p>	<p>12.31 seconds</p>
<p>Test case 4</p>  <p>Search status - Goal Reached!</p> <p><i>Figure 4.6, Chapter 4, test case 4 for optimal path</i></p>	<p>14.74 seconds</p>
<p>Test case 5</p>  <p>Search status - Goal Reached!</p> <p><i>Figure 4.7, Chapter 4, Test case 5 for optimal path</i></p>	<p>15.59 seconds</p>

Table 4.1, Chapter 4, table depicts the time taken from source to destination in different maps

4.3 Conclusion

The result obtained is used to find the shortest path from one point to another point, we can trace the path covered before it actually reaches the destination. In this given scenario we made a complex unknown environment with static obstacles to reach the destination without any problems. A star algorithm performs much better to find the optimal path in all these given scenarios or map.

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Brief summary of the work

A* is often or usually used for the familiar pathfinding problem in applications such as simulation games but was originally sketched as a general graph traversal algorithm. It finds applications in various problems. Other cases involve an Informational search with online learning. We have discussed on how A star search algorithm will be used to make a pathfinder and how it helps us to find the shortest or optimal path from source or initial point to destination or final point.

5.1.1 Problem statement in brief

We implemented A star search algorithm to find the shortest path between nodes or graphs and use this as a pathfinder. It is observed that it is an informed search algorithm, as it uses information about path cost and also uses heuristics to find the solution.

The secondary objective is to enhance a drive less car by using path tracking algorithm. A human driver or operator drives the given path initially, while a computer is constantly recording the position or location , velocity or speed, orientation or direction, and steering angle. These details are basically used to command the vehicle whenever it autonomously travels along the given trail. On condition that the vehicle gets astray, for instance, as a result of keeping away from an obstacle or due to noise produced within the positioning sensors. We find the simplest possible actions or path it should absorb a selected situation with the aid of this algorithm.

5.1.2 Work methodology adopted

A* is often or usually used for the familiar pathfinding problem in applications such as simulation games but was originally sketched as a general graph traversal algorithm. The given algorithm has 3 variables which are explained below:

- 1) F – F is the variable of A star which is the sum of the other variables H and G and is the minimum or least cost from one node to the succeeding or next node. This variable is in charge for guiding us to find the most optimal path from our starting point or source to final point or destination.
- 2) H – H is the given variable for heuristic/estimated path between the actual or current node to the final point or destination node. This cost is not genuine but is, in reality, a estimate cost will be used to find the most optimal path between our source or starting point and destination or final point.
- 3) G – G is the given variable that depicts cost of node from one node to the other subsequent node. This variable changes for every node as we move towards upward direction to find the most optimal path.

So, from the above we can say $F = G + H$.

The A star search algorithm is widely used for path finding and also graph traversal. The given algorithm efficiently plots an optimal path between multiple nodes or points on the given graph. But A star is slow and also takes a lot of space as it saves all the available paths that are feasible to us. This makes other quicker algorithms have an upper hand over A star, but it is nevertheless, one of the best algorithms out there.

5.2 Conclusion

5.2.1 General conclusion

Some of the advantages and disadvantages of A star search algorithm [15] can be explained below:

Pros:

- 1) It will always find the ideal solution to be implemented that it exists and that if a heuristic is provided then it must be allowable.
- 2) Heuristic is unnecessary; it is used to accelerate the task.
- 3) Various heuristics can be unified to the algorithm without changing the basic code.
- 4) The cost of each move can be modified into the algorithms as easily and smoothly as the heuristic.
- 5) It is not constrained to a unidirectional search.

Cons:

- 1) Not the finest algorithm for each problem in terms of processing required and Memory.
- 2) usage of lot of memory since each node made has to be kept accounted for.

5.2.2 Significance of the result obtained

The result obtained in A star algorithm is used to find the shortest path from one point to another point, we can trace the path covered before it actually reaches the destination. In this given scenario as explained in the methodology 3.2.3 we made a complex unknown environment with static obstacles to reach the destination without any problems. A star algorithm performs much better to find the optimal path in all these given scenarios or map.

5.3 Future Scope of work

1) Use of A star algorithm in games: In games we often want to find paths from one location to another. We are not only trying to find the shortest distance; we also want to consider travel time. Understanding on how to get from point A to point B is [16] something many games require. Whether you are designing a deep by-turn strategy tactics RPG or a easy puzzle game, exploring your game world often needs more intelligence from your game characters than merely pointing towards a final point and moving in a straight line.

You would not want the characters in your game to walk or pass through walls. Nor would players allow characters to get stuck and bump into walls when a way around an bump is clearly obvious. There's nothing more annoying than a real time strategy game's unit being not able to direct their way to a target location.

This is where pathfinding algorithms like A star comes in. Pathfinding is the basic element of most game A.I.

2) Use of A star algorithm for train travel management system [17]: The A star algorithm deals with rearranging the process of train travel management system. In order to so decrease the delays of train when natural disasters, accidents or technical problems occur on railway lines.

A rescheduling algorithm of train groups to reschedule trains, instead of previous just waiting. A Petri-net-based conflict resolution algorithm adopted from the A-star algorithm is designed to find for an ideal or a near-flawless realistic schedule. The algorithm acknowledge the railway working principles when generating new characterizing, so that the new schedule has less train delays and respects the safety principles.

An evaluation principle based on the extent of the interference towards the original timetable and in a particular time is designed to select the optimal path for each train. Show that the rescheduling algorithm is able to reduce the delays of train time and make the parallel processing, make a contact of the real-time response of the train-group rescheduling well.

References

- [1]Artificial Intelligence, <https://www.investopedia.com/terms/a/artificial-intelligence-ai.asp>
- [2]Machine learning, https://en.wikipedia.org/wiki/Machine_learning
- [3]Applications of artificial intelligence, <https://www.valluriorg.com/blog/artificial-intelligence-and-its-applications/>
- [4]Applications of machine learning, <https://www.javatpoint.com/applications-of-machine-learning>
- [5]Tingxin Yan, Baik Hoh, Deepak Ganesan, Ken Tracton, Toch Iwuchukwu and Juong-Sik Lee. A crowdsourcing based parking booking system for mobile phone devices. Technical Report UM-CS-2011-001, Department of CSE, University of Massachusetts, Amherst MA,2011
- [6]A star search algorithm, https://rosettacode.org/wiki/A*_search_algorithm
- [7]termination and completeness in A star algorithm, https://en.wikipedia.org/wiki/A*_search_algorithm#Termination_and_Completeness
- [8]A star search algorithm, <https://www.geeksforgeeks.org/a-search-algorithm/>
- [9]Indian Auto Industry , <http://www.knowindia.net/auto.html>
- [10]Amidi, O. (1990), “Integrated Mobile Robot Control”, Carnegie Mellon Univ. Robotics Institute, Technical Report CMU-RI-TR-90-17, Pittsburgh, Pennsylvania, (USA).
- [11]Follow the past algorithm ,https://people.cs.umu.se/thomash/reports/07_Hellstrom.pdf
- [12]Implementation of pure pursuit path tracking algorithm, https://www.ri.cmu.edu/pub_files/pub3/coulter_r_craig_1992_1/coulter_r_craig_1992_1.pdf

[13]follow the carrot algorithm, https://www.researchgate.net/figure/Carrot-Following-Approach_fig3_3955553

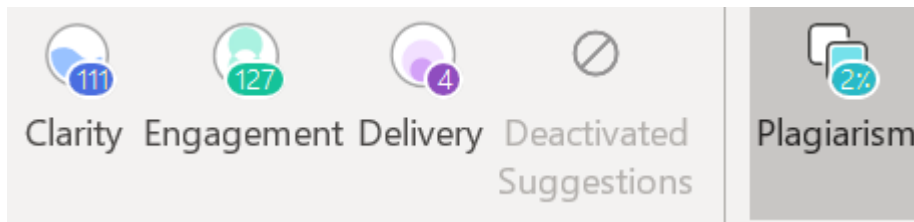
[14]working of A star algorithm, [edureka.co/blog/ a-search-algorithm/](http://edureka.co/blog/a-search-algorithm/)

[15]A-star in general, <https://algorithmsinsight.wordpress.com/graph-theory-2/a-star-in-general/>

[16] A-STAR Pathfinding AI for HTML5 Canvas Games , <http://buildnewgames.com/astar/>

[17] Use of A star algorithm for train travel management system ,<http://www.ijptonline.com/wp-content/uploads/2017/01/19321-19327.pdf>

PLAGIARISIM REPORT



Plagiarism level: 2%

Software used: Grammarly premium

PROJECT DETAILS

<i>Student Details</i>			
Student Name	Vishal Vimal Kumar		
Register Number	160905071	Section / Roll No	C/21
Email Address	Vishalvimal3321@gmail.com	Phone No (M)	+91-7406290094
<i>Project Details</i>			
Project Title	Smart car parking using pathfinder		
Project Duration	4 months	Date of reporting	06-01-2020
<i>Organization Details</i>			
Organization Name	Manipal Institute of Technology		
Full postal address with pin code	Udupi - Karkala Rd, Eshwar Nagar, Manipal, Karnataka 576104		
Website address	https://manipal.edu/mit.html		
<i>Internal Guide Details</i>			
Faculty Name	Archana Praveen Kumar		
Full contact address with pin code	Dept of Computer Science & Engg, Manipal Institute of Technology, Manipal – 576 104 (Karnataka State), INDIA		
Email address	archana.kumar@manipal.edu		