Path Planning of Autonomous Vehicles Based on Artificial Potential Field Method

Hanxue Cheng

Shanghai Ulink Bilingual School, No.559, Laiting South Road, Shanghai, 519000, China
Corresponding Author: Hanxue Cheng, Email: 1031676079@qq.com

Abstract

Technology is developing rapidly nowadays. As automation of vehicles is becoming increasingly mature, it has been a topic of focus. At the same time, the automotive industry is seen to be making a monumental paradigm shift from manual to semi-autonomous to fully autonomous vehicles. As such, this paper attempts to offer an approach for path planning of an autonomous vehicle based on the artificial potential field method. To evaluate the performance of the proposed approach, the algorithm is verified on MATLAB R2021b, by assuming that the autonomous vehicle is a particle. The result shows that this approach can basically complete the obstacle avoidance function. However, this approach inevitably encounters some problems in some cases. In the search for better performance of this function, we adjusted some environmental parameters and find out the effect of these changes. At the end of the paper, we conclude with solutions corresponding to several possible problems, according to the results from the previous search and some articles.

Keywords

Autonomous vehicles; Path planning; Artificial potential filed method; Obstacle avoidance

Introduction

Distracted driving has become one of the major reasons for the increasing number of road accidents worldwide. According to a study by National Highway Traffic Safety Administration (NHTSA), in the US alone, 3477 people were killed and 391,000 were injured due to distracted driving in 2015 (NHTSA, 2015). These compelling statistics demonstrate the significant necessity of evolution in the automotive industry to migrate away from traditional manual driving to different levels of vehicle automation step by step. It is believed that through this migration, artificial intelligence (AI) intervenes in. The advantage of artificial intelligence (AI) over humans is its great ability to real-control in a dynamic environment, allowing it to make accurate judgments about the environment in a very short time. Thus accidents can be avoided and the possibility of distracted driving can be eliminated.

The History of Autonomous Vehicles

Historical advancements through time have had a major impact on the currently available autonomous technologies. This section will focus on some of those major landmarks (Bimbraw, 2015). One of the very first autonomous cars was built in 1926 and was

Citation: Hanxue Cheng. (2022) Path Planning Of Autonomous Vehicles Based On Artificial Potential Field Method. The Journal of Young Researchers 4(2): e20220520

Received on May 7, 2022; Accepted on May 10, 2022; Published on May 20, 2022

called the Linriccan Wonder (Bimbraw, 2015). The operating principle was very simple. It consisted of a transmitting antenna that captured radio signals from another car that would be following it. The car's motion was then controlled by small electric motors which were in turn connected to circuit breakers receiving signals from an antenna. In December 1926, Achen Motors showcased a modified version of the Linriccan Wonder, in Milwaukee, which was known as the Phantom Auto.

The year 1939 could be marked as another major stepping stone in the development of autonomous vehicles. Electric cars which were powered using embedded circuits were presented at the world fair by Normal Bel Gedde (Bimbraw, 2015). General Motors had sponsored their Futurama exhibit. Following this, in 1953, RCA labs had built a small autonomous car that would be controlled based on a pattern of wires. General Motors in collaboration with Leland Hancock and L.N. Ress were able to develop this idea further and take it to the actual road. As a result, Firebird, which consisted of a series of experimental semi-autonomous vehicles was launched in the General Motors Auto show called Motorama in the 1960s (Cranswick, 2013; Burgan, 1999; Temple and Adler, 2006). This period had already marked successful simulations of primary vehicle controls such as automatic braking, accelerating, and steering. These vehicles mainly worked with the help of devices which were installed within the roadway in order to guide the vehicles.

Inspired by the Firebird, in 1966, a similar driver-less car was developed by the Communication and Control Systems Laboratory team at the Ohio State University. This technology of embedded devices within the roadway guiding the vehicle had become very popular in the 1960s. Following a similar concept, vehicles were also controlled by embedded magnetic cables. Citroen DS is one such example of an autonomous car developed by the Transport and Road Research Laboratory in the UK (Cardew, 1970; Pressnell, 1999).

The concept of vehicle automation was well supported by the Bureau of Public Roads in the USA as well with their experimental initiative of an electronically controlled highway, which was well supported by states such as California, Massachusetts, New York, and Ohio (Toyota, 2016). Following this, Bundeswehr University Munich also developed an autonomous van sponsored by Mercedes. In addition, the Prometheus project conducted by EUREKA during the period of 1987 – 1995 also gained a lot of importance in the field of autonomous vehicles (Ming et al., 1993; Flyte, 1995). Some of the other autonomous vehicle projects during this period included the ones by the US Department of Defense's Defense Advanced Research Projects Agency, Carnegie Mellon University, the Environmental Research Institute of Michigan, the University of Maryland, Martin Marietta, and the SRI International (Bimbraw, 2015). The combined project with the various universities was known as the Autonomous Land Vehicle Project (Davis et al., 1987; Leighty, 1986; Lowrie et al., 1985; Chandran et al., 1986).

Furthermore, various political changes mainly in terms of the ISTEA Transportation Authorisation Bill of 1991 in addition to the establishment of the National Highway System Consortium in US became major stepping stones in the development of driver-less vehicles. Daimler – Benz's VaMP, Bundeswehr University Munich's Vita-2, Dickmanns' driver-less S-Class Mercedes, Carnegie Mellon University's Navlab, more popularly called No Hands Across America or NHOA, and Alberto Broggi's ARGO were some of the major exciting successful autonomous-capable vehicle projects during the period of 1990s (Behringer and Muller, 1998; Wenger, 2005; Franke et al., 1997; Thorpe et al., 1991; Pomerleau, 1993; Broggi et al., 2000).

Finally, more sophisticated and efficient designs of autonomous vehicles were explored in 2000s (Bimbraw, 2015). Various off-road, military as well as public transportation options were evaluated for the implementation of autonomous capabilities. One prominent example is a public ground transportation system called ParkShuttle, which was implemented in the Netherlands (Shladover, 2007; Panayotova, 2003; Andréasson and passion, 2001). United States' efficient military demo vehicles became exemplars in demonstrating the application of autonomous vehicles (Hong et al., 2000).

Typical Autonomous Vehicle System

This section provides an overview of the various stages involved in an ADAS architecture. Figure 1 depicts the major steps involved in the functionality of an autonomous-capable vehicle from a high level perspective (Kato et al., 2015). Autonomous vehicle actions can be classified into three broad categories – see, decide and execute (Patchett, 2015) .

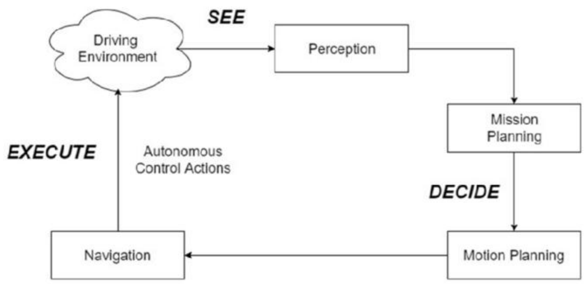

Figure 1. High-level ADAS architecture

Perception: The perception module helps to accurately sense the driving environment of the autonomous vehicle. Simultaneous Localization and Mapping is one of the most important and primary steps (Kato et al., 2015). The autonomous vehicle has to be first sensed and accepted into its driving environment. Then an accurate real-time 3D map could be generated and updated frequently as the vehicle goes through its drive cycle. As described in Kato et al. (2015), a three dimensional Normal Distributions Transform algorithm (Magnusson et al., 2007) can be used in order to perform accurate matching and mapping operations with the 3D cloud data.

Following the localization step, the Object Detection stage needs to be taken care of. The autonomous vehicle needs to detect all kinds of objects including moving vehicles and pedestrians, motorcycles, bicycles, animals, stationary vehicles and pedestrians, traffic signals, stop signs and the other dynamic and static objects present in the driving environment. In order to reduce the computation cost, it is very important to focus the object detection only on a specific area or road which is of interest. As specified in Kato et al. (2015), the Deformable Part Models Algorithm can be used for accurate object detection (Felzenszwalb et al., 2010). In McNaughton (2011), to increase computational efficiency, parallel algorithms implemented on graphics processing units (GPU) are designed to accelerate the information processing.

The next step is called Object Tracking (Kato et al., 2015). In addition to detecting where a particular object is in the concerned scope of space on the road, it is also vital to track the changes in their position. In other words, the motion of the objects needs to be detected and updated in real time. This will provide the autonomous vehicle with a better understanding of its dynamic environment, which in turn will be helpful in updating its intelligent control as part of a closed-loop control system.

Projection and Re-projection are the next steps in the process of accurate perception (Kato et al., 2015). Once the objects' changing position has been detected using the various sensors and the cameras, the information needs to be summarized and combined in order to get a better depth estimate of the interested space on the road. In addition, this information can be used to further update the three-dimensional cloud data in order to have real-time path updates.

Mission planning: Once the path is finalized, the Mission Planning step (Kato et al., 2015) could be initialized where the required action of the autonomous vehicle is estimated based on the required functionality. For example, in a Park Assist or Self Parallel Parking functionality, the autonomous vehicle could take over and accurately park the car depending on the parking boundaries detected. In an Adaptive Cruise Control strategy, the mission planner could be helpful in planning the required motion of the vehicle based on the dynamic traffic and surroundings. This paper will focus on the implementation of path planning, which belongs to this mission planning module.

Motion planning: Following Mission Planning then, is the Motion Planning step (Kato et al., 2015). In other words, once the mission or the destination is planned, the motion or the maneuver which needs to be performed by the autonomous vehicle in order to accurately and optimally (time, cost and fuel) reach the programmed destination needs to be planned. The use of conformal spatial and temporal lattices (McNaughton et al., 2011) for planning the motion according to the changes in the environment has been demonstrated in Kato et al. (2015).

Navigation: Navigation or Path Following is the concluding step (Kato et al., 2015). Based on a planned path, using the previous steps, the autonomous vehicle controller aids in keeping the vehicle on the desired path for the entire trip. As suggested in Kato et al. (2015), the Pure Pursuit Algorithm can be used for the path following step (Coulter, 1992). The aim is to divide the path into manageable milestones. An error calculation would be done in real-time in order to make sure that the vehicle is on the desired trajectory at every milestone. In case of a significant deviation, the control and tracking input would be updated accordingly in order to cover the difference. A smaller duration in between the milestone checks would result in more accurate vehicle control.

Different Levels of Vehicle Automation

When it comes to autonomous vehicles, there is a common misconception that an autonomous vehicle can only refer to a completely driverless vehicle. However, that is not the complete story. There are various levels of autonomy that can categorize these vehicles. According to the Society of Automotive Engineers (SAE) J3016 standard, there are six distinct levels of Vehicle Automation (SAE, 2014).

These levels are categorized based on four parameters – control of lateral and longitudinal vehicle motion, Object and Event Detection and Response (OEDR), Dynamic Driving Task (DDT) and Operational Design Domain (ODD). OEDR mainly consists of the perception, response formulation and reaction with an autonomous vehicle. DDT includes all of the vehicular activities performed that contribute to a particular motion of the vehicle. This does not include any planning type of strategic activities (SAE, 2014). ODD refers to a scope of conditions in which the vehicle has been designed to operate in. Depending on how the responsibility for these 4 categories is allocated between a human driver vs. the Automated Driving System (ADS), a level of vehicle autonomy is assigned.

These six levels can be summarized in a table, as shown below.

Table 1. Different levels of Vehicle Automation

Level	Name	Control of Lateral and Longitudinal Vehicle Motion	Object and Event Detection and Response (OEDR)	Dynamic Driving Task (DDT)	Operational Design Domain (ODD)
Level 0	No Driving Automation	Driver only	Driver and System	Driver only	limited
Level 1	Driver Assistance	Driver and System	Driver and System	Driver only	limited
Level 2	Partial Driving Automation	System only	Driver and System	Driver only	limited
Level 3	Conditional Driving Automation	System only	System only	Dynamic Driving Tasks Take over the user	limited
Level 4	High Driving Automation	System only	System only	System only	limited
Level 5	Full Driving Automation	System only	System only	System only	unlimited

Level 0: Level 0 (No Driving Automation) is the first level where the driver is mainly responsible for all the major control of the vehicle consisting of steering, braking and throttle. At this level, although the driver is taking the responsibility for the entire DDT, other active safety systems may be present in the vehicle. Blind-spot monitoring, collision warning and lane departure warning systems are some examples of level 0 automation.

Level 1: Level 1 (Driver Assistance) is the second level of vehicle automation. At this level, the driver is still mainly responsible for the overall vehicle control. However, level 1 automation allows for the use of an automated system that may support the driver in only one of the lateral or longitudinal vehicle motion control. The responsibility for the OEDR and DDT fallback is still on the driver. Some examples of level 1 automation include adaptive cruise control, electronic stability control, automatic braking and lane-keeping.

Level 2: The next level of autonomy is Level 2 (Partial Driving Automation). In this level, the ADS is permitted to take control of both the lateral and the longitudinal vehicle motion. However, the driver is still expected to supervise the ADS and take responsibility for the OEDR and DDT fallback.

Level 3: The fourth level of autonomy is referred to as Level 3 (Conditional Driving Automation). At this level, the autonomous system is able to take primary control of the vehicle, including the lateral and longitudinal control and the OEDR, ensuring a safe operation. However, it is advisable for the driver to be present in case a switch of the operation mode is intended by the ADS from autonomous to driver-controlled. As such, the DDT fallback still remains as the human driver, adding additional layers of complexity, unlike the other levels lower and higher than this.

Level 4: The fifth level of vehicle automation is termed as Level 4 (High Driving Automation). In this level of vehicle automation, the automated system is expected to take full control of the vehicle with no intervention expected from the driver. In other words, the system is responsible for the OEDR, the vehicle's full control and the DDT fallback. However, the ODD is still expected to be limited, unlike Level 5, which is the final level.

Level 5: In Level 5 (Full Driving Automation), the ADS is again expected to take full responsibility for the entire vehicle control, OEDR and the DDT fallback, however, the ODD is no longer limited to a specific design domain scope.

Method

For the realization of automation of vehicles, we raise a method called the artificial potential field method. The artificial potential field method is widely used for autonomous vehicle path planning due to its efficient mathematical analysis and simplicity. Path planning refers to the fact that the robot finds a collision-free optimal path from the starting point to the target point in the workspace with obstacles, on the premise of one or more optimization criteria among the shortest moving path, the shortest moving time and the minimum working cost. Numerous algorithms for autonomous vehicle path planning have been developed. Herein, we propose an improved artificial potential field method base on gravity chain that connects with the beginning and the ending. To evaluate the performance of the proposed method, by assuming that the autonomous vehicle is a particle, the algorithm is verified on MATLAB R2021b. In the discussion section, we also investigate the influence of the parameters of the environment and goals to the path planning of autonomous vehicles.

The Basic Principles of Artificial Potential Field Method

Let us illustrate the mechanism of action of the artificial potential field method with two analogies. First, we compare the configuration space to a potential field plane and the robot (of the current configuration) to a point in the space. If we let the starting point of the robot and the obstacle have a positive charge, the endpoint has a negative charge and the robot has a positive charge. Due to the principle that the same charges repel and opposite charges attract, the robot will move along a path towards the endpoint under the force of the electric field and avoid the positively charged obstacles, as shown in Figure 2.

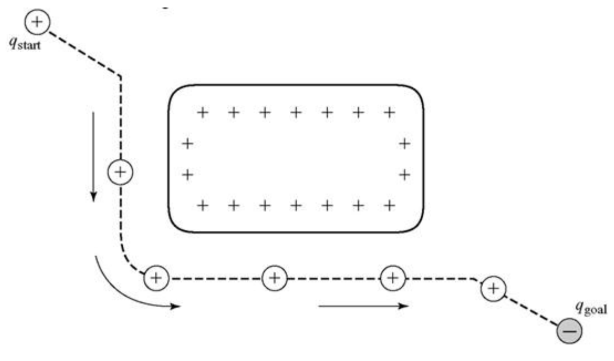

Figure 2. Electric potential field

Similarly, we can compare the configuration space to a region with undulating terrain. Where the starting point and obstacles are located in the higher area while the end point is located in the lower area, the robot is treated as a sphere. Then under the effect of gravity, the robot will slide down a certain trajectory from the higher starting point to the lower ending point and avoid the higher obstacles. This is shown in Figure 3.

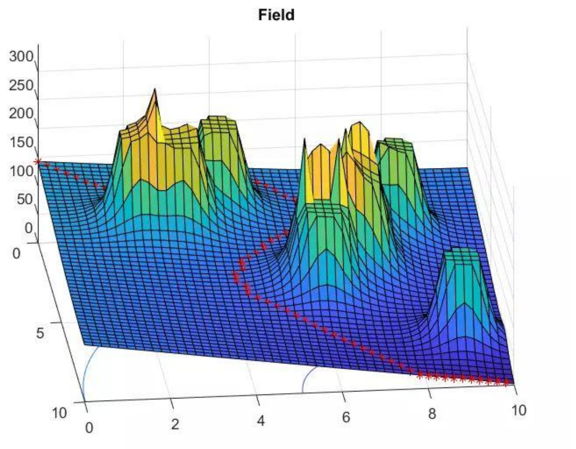

Figure 3. Gravitational potential field
(image from Ref. [14])

The two examples above are the mechanism of action of an electric potential field and a gravitational potential field, both of them are natural potential fields. The artificial potential field method involves constructing an artificial potential field to mimic this mechanism of action when the starting point, endpoint and obstacle locations are known. The advantage of the artificial potential field method is that it is in fact a feedback control strategy that is robust to control and sensing errors; the disadvantage is that there is a local minima problem and therefore there is no guarantee that a solution to the problem will be found.

Main Program

The main program is started by initializing parameters of the artificial potential field algorithm, including the gravitational gain of the attractive potential function, the repulsion gain of the repulsive potential function, and the distance threshold of the obstacle's action (greater than which the obstacle does not exert a repulsive influence). Then initialize the starting point and the target point. The environment model is subsequently demonstrated to obtain the position of obstacles and extend the obstacle's border by their length and width. The position and size of the obstacles are randomly generated.

In the calculation part of the main program, the angle calculation module, the attractive potential calculation module, the repulsive potential calculation module and the gradient descent calculation module were called. All of these calculations can decide the next position that the autonomous vehicle will move to.
At last, the path generated by the proposed method for autonomous vehicles is shown. The result can be illustrated in static or dynamic images.

Computation of Angles

This is expressed in program code as follows:

```
Function

 Y=compute_angle(X,Xsum,n)%Y is the angle vector between the x-axis and the attractive force, the repulsive force. X is the coordinate of the starting point , X sum is the coordinates vector of the target point and the obstacles, which is a (n+1)*2 matrix.
```

```
for i=1:n+1

    deltaX(i)=Xsum(i,1)-X(1);

    deltaY(i)=Xsum(i,2)-X(2);

    r(i)=sqrt(deltaX(i)^2+deltaY(i)^2);

    if deltaX(i)>0

        theta=acos(deltaX(i)/r(i));

    else

        theta=pi-acos(deltaX(i)/r(i));

    end

    if i==1

        angle=theta;

    else

        angle=theta;

    end

Y(i)=angle;

%Save every angle in Y vector. The first element is the angle to the target point, others are the angles to the obstacles

end

end
```

Computation of Attractive/ Repulsive Potential

We use the potential function U to create artificial potential fields. A potential (field) function is a differentiable function, and the magnitude of the potential function at a point in space represents the strength of the potential field at that point. The simplest potential function is the attractive/repulsive potential function. The idea behind its action is simple: make the target attractive to the robot and the obstacle repulsive to the robot. The potential function U(q) at a point is expressed as the sum of the attractive and repulsive potentials:

$$U(q) = U_{att}(q) + U_{rep}(q)$$

One of the most common expressions for the gravitational potential function is as follows:

$$U_{att}(q) = \frac{1}{2}\zeta d^2(q, \quad _{goal})$$

ζ - gravitational gain

$d^2(q, q_{goal})$ - distance from current point q to target point q_{goal}

The most common expressions for the repulsive potential function are as follows:

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta(\frac{1}{D(q)} - \frac{1}{Q^*})^2, & D(q) \le Q^* \\ 0 & D(q) > Q^* \end{cases}$$

$D(q)$ - the distance of the point q from its nearest obstacle

η - repulsion gain

Q^* - the distance threshold of the obstacle's action, greater than which the obstacle does not exert a repulsive influence

Of course, the design of the above gravitational and repulsive potential functions can be problematic in some cases, so there are many ways to improve them, which we will discuss later.

Computation of Gradient Descent

If the value of the potential function U(q) at a point q is taken to be the energy level at that point, then the gradient $\nabla U(q)$ can be considered as the force vector at that point, defined as

$$\nabla U(q) = DU(q)^T = \left[\frac{\partial U}{\partial q_1}(q), \ldots, \frac{\partial U}{\partial q_m}(q)\right]^T$$

It can be seen that the direction of the gradient at a point is the direction in which the potential function grows fastest.

The gradient descent method, on the other hand, allows the robot to start at the initial point and keep walking in the opposite direction of the gradient until the gradient is 0. This is expressed in pseudo-code as follows:

```
Input： A method for calculating the gradient at point q∇U (q)

Output：A sequence if trajectories {q(0), q(1), ..., q(i)}

q(0) = q_start

i = 0

while ∇U (q(i)) ≠ 0 do

    q(i + 1) = q(i) + α(i)∇U (q(i))

    i = i + 1

end while
```

Results

The path planning result is shown in Figure 4.

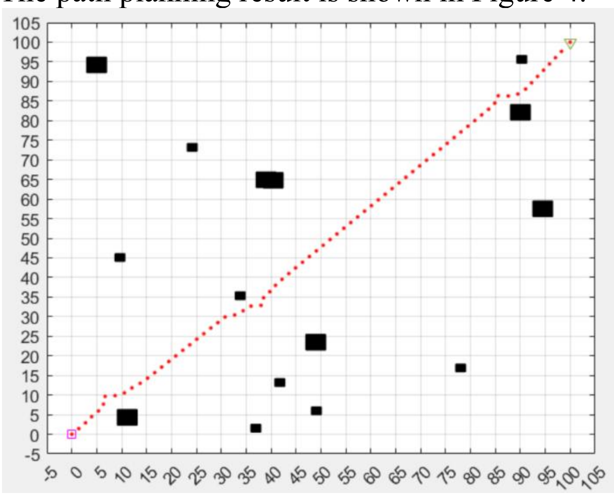

Figure 4. Results of path planning by improved Artificial Potential Field Method

The figure shows that the autonomous vehicle staring from the origin and follows the shortest path to the target point. When the autonomous vehicle encounters obstacles, it redirects its route to bypass the obstacles. The simulation results show that it is a simple and effective method for obstacle avoidance and path planning of autonomous vehicles.

Influence of the Position Distribution of Obstacles

In the code, the variable a is used to control the distribution region of the obstacles. By setting a different value for a, the distribution region of the obstacles will be different.

Starting from a = 100, the position of obstacles is distributed throughout the entire region [0-100, 0-100]:

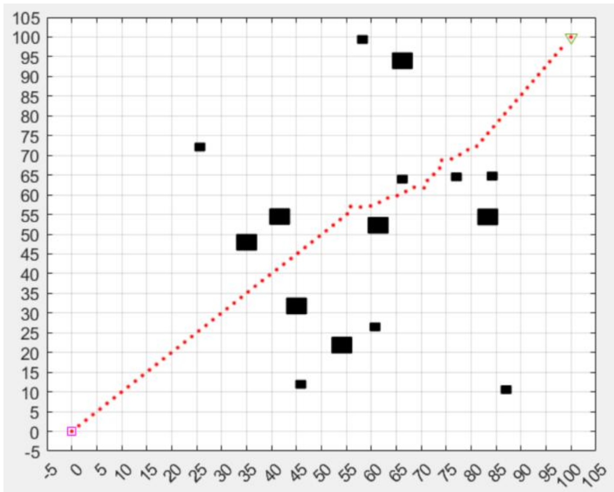

Figure 5. Results of path planning where the position of obstacles is distributed throughout the region [0-100, 0-100]

The image shows that all obstacles are randomly generated and distributed in the given area. The autonomous vehicle travels in a straight line through the region with no influence of obstacles. When it encounters a region of the congregation of obstacles, it redirects its route to bypass the obstacles and finally reaches the target point.

When a = 50, the position of obstacles is distributed in the region [0-50, 0-50]:

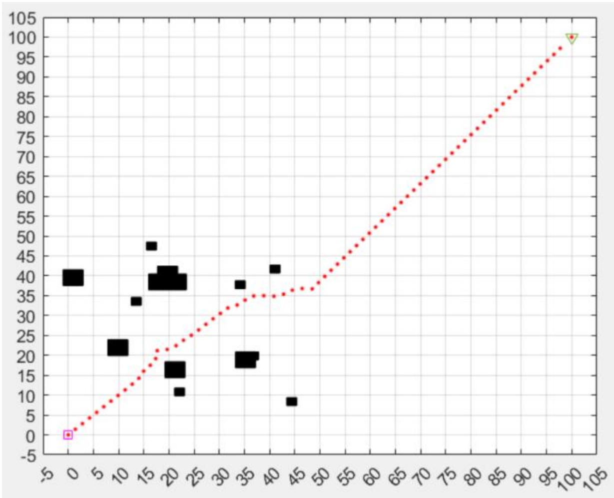

Figure 6. Results of path planning where the position of obstacles is distributed throughout the region [0-50, 0-50]

The image shows that all obstacles are randomly generated and distributed in the given area. When the autonomous vehicle encounters a region of the congregation of obstacles [0-50, 0-50], it redirects its route to bypass the obstacles. Then in the region [50-100, 50-100], its path is a straight line and finally reaches the target point.

When a = 25, the position of obstacles is distributed in the region [0-25, 0-25]:

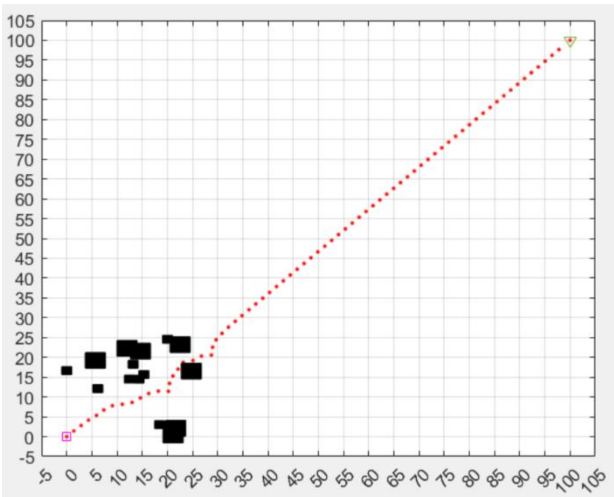

Figure 7. Results of path planning where the position of obstacles is distributed throughout the region [0-25, 0-25]

The image shows that all obstacles are randomly generated and distributed in the given area. When the autonomous vehicle encounters a region of the congregation of obstacles [0-25, 0-25], it redirects its route to bypass the obstacles. However, there is a risk of hitting the obstacles because the obstacles are too dense. The gravitational gain may need to be adjusted to avoid this kind of situation. Then in the region [25-100, 25-100], its path is a straight line and finally reaches the target point.

Influence of the Number of Obstacles

In the code, the variable n is used to control the number of obstacles. By setting a different value for n, the number of the obstacles will be different.

When n = 5, the number of obstacles is 5:

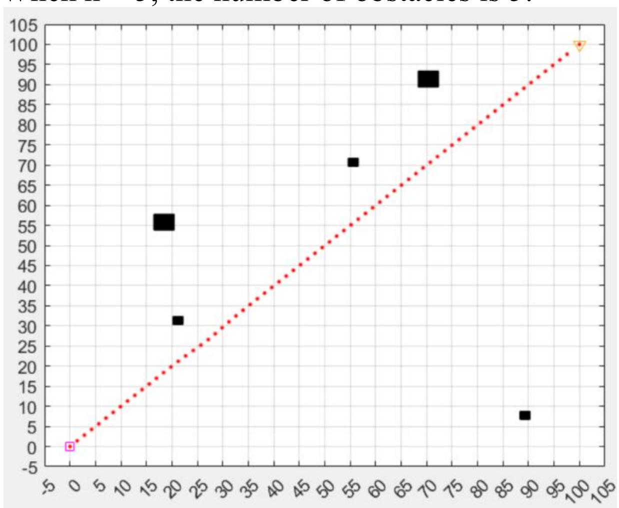

Figure 8. Results of path planning where the 5 random obstacles are distributed throughout the entire region

The image shows that all the 5 obstacles are randomly generated and distributed in the whole region. However, they almost do not affect the path of the autonomous vehicle because they are too sparse.

When n = 10, the number of obstacles is 10:

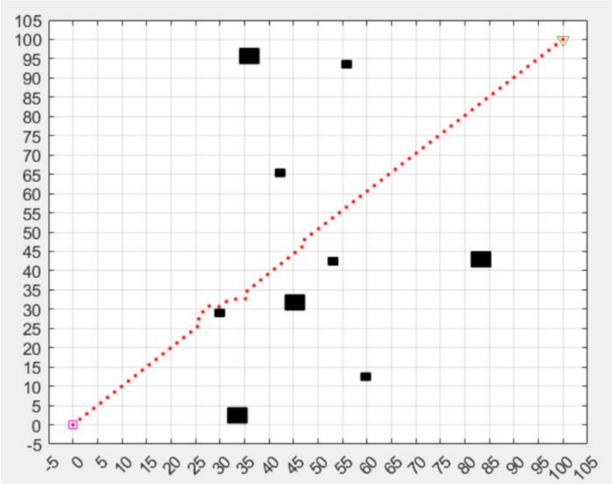

Figure 9. Results of path planning where the 10 random obstacles are distributed throughout the entire region

The image shows that all the 10 obstacles are randomly generated and distributed in the whole region. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles.

When n = 20, the number of obstacles is 20:

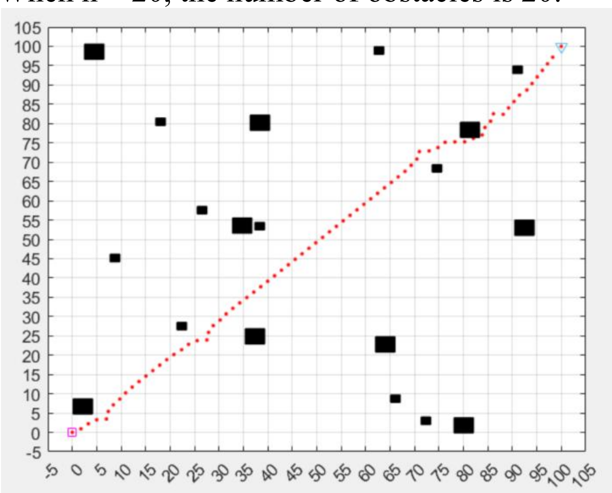

Figure 10. Results of path planning where the 20 random obstacles are distributed throughout the entire region

The image shows that all the 20 obstacles are randomly generated and distributed evenly in the whole region. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles.

When n = 30, the number of obstacles is 30:

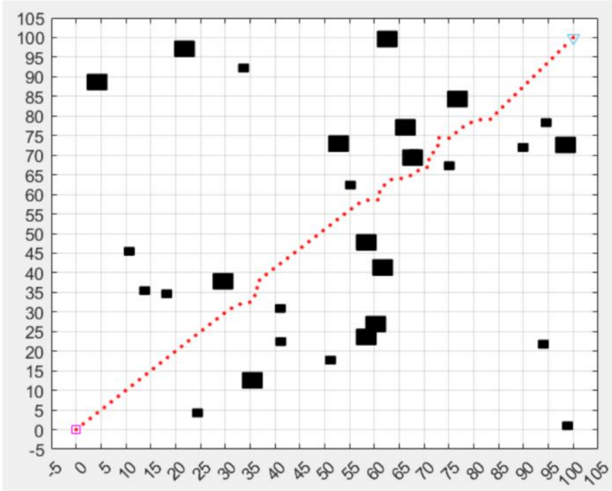

Figure 11. Results of path planning where the 30 random obstacles are distributed throughout the entire region

The image shows that all the 30 obstacles are randomly generated and distributed in the whole region. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. However, there is a risk of it hitting the obstacles because the obstacles are too dense. The gravitational gain may need to be adjusted to avoid this kind of situation.

Influence of the Starting Point

In the code, the array Xo stores the x, and y coordinates of the starting point. By setting a different value for the x,y coordinates, the autonomous vehicle will start the driven route from different points.

When it starts at Xo = [0,0], the origin:

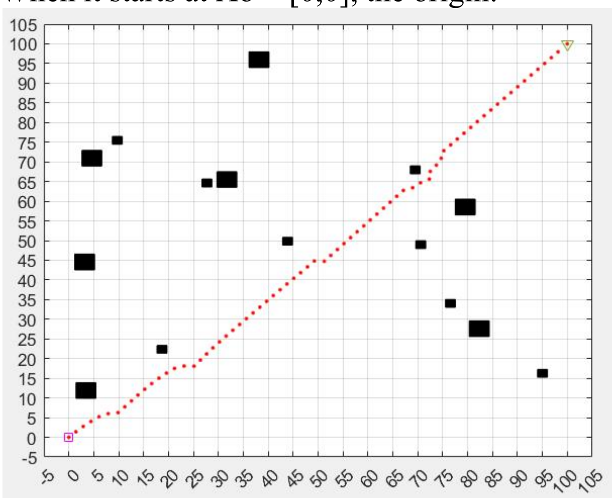

Figure 12. Results of path planning where the starting point is (0,0) and the target point are (100,100)

The image shows that the autonomous vehicle starts from point [0,0] and reaches the target point. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles.

Change the x-coordinate, the starting point is Xo = [50,0]:

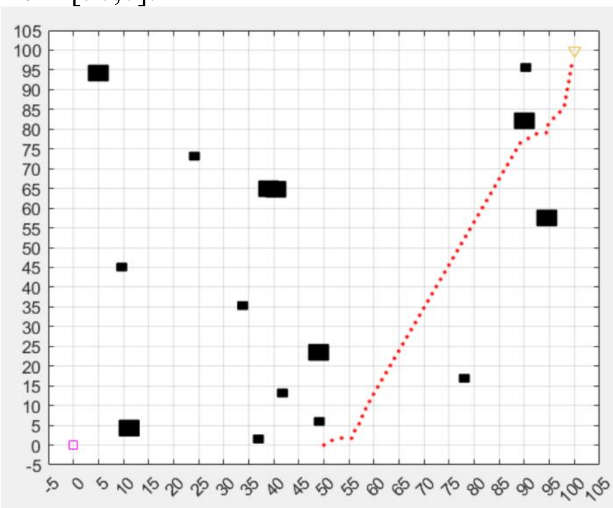

Figure 13. Results of path planning where the staring point is (50,0) and the target point is (100,100)

The image shows that the autonomous vehicle starts from point [50,0] and reaches the target point. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. The whole driven route is steeper than before.

Change the x-coordinate, the starting point is Xo = [90,0]:

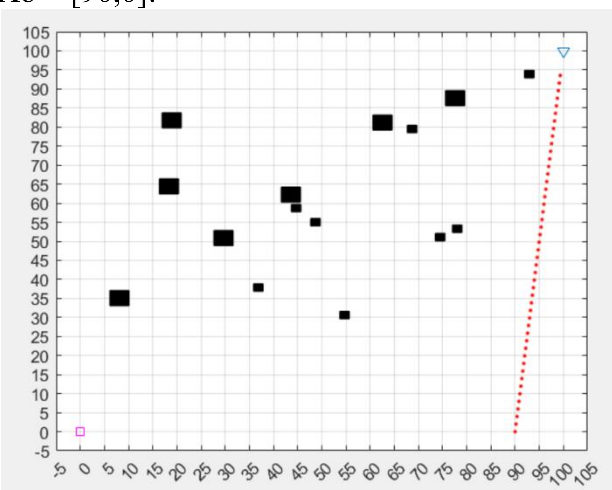

Figure 14. Results of path planning where the starting point is (90,0) and the target point is (100,100)

The image shows that the autonomous vehicle starts from point [90,0] and reach the target point. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. The whole driven route is very steep. According to the attractive function, when d (the distance between the starting point and the target point) becomes smaller, the driven route depends on the repulsive function more. That`s the reason why the closer between the starting point and the target point, the more effective for the autonomous vehicles to bypass the obstacles.

Influence of the Target Point

In the code, $X_{sum}(1,1)$ stores the x-coordinate of the target point and $X_{sum}(1,2)$ stores the y-coordinate of the target point. By setting different values for them, the autonomous vehicle will end the driven route at different points.

Change the y-coordinate, the target point is [100, 50]:

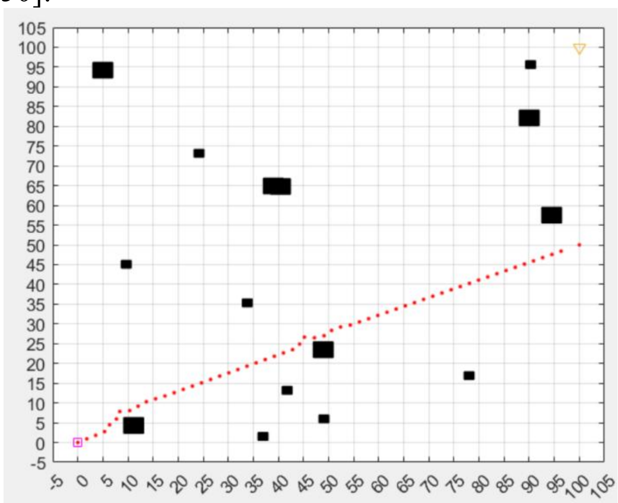

Figure 15. Results of path planning where the staring point is (0,0) and the target point is (100,50)

The image shows that the autonomous vehicle starts from the origin and reaches the target point [100, 50]. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. The whole driven route is gentler than before.

Change the y-coordinate, the target point is [100,30]:

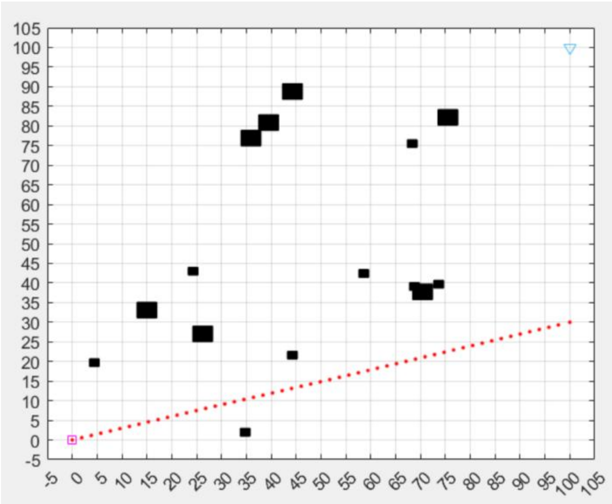

Figure 16. Results of path planning where the starting point is (0,0) and the target point is (100,30)

The image shows that the autonomous vehicle starts from the origin and reaches the target point [30, 100]. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. The whole driven route is gentler than before.

Change the coordinates of both the two points, the starting point is [0,20] and the target point is [100,30]:

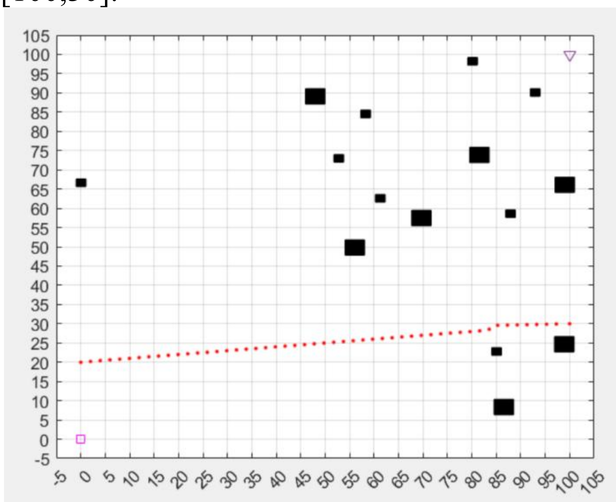

Figure 17. Results of path planning where the starting point is (0,20) and the target point is (100,30)

The image shows that the autonomous vehicle starts from the point [0,20] and reaches the target point [100,30]. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. The whole driven route is very gentle.

Influence of the Distance Between Starting Point and Target Point

Here is one special case.

Change both x, y coordinates, the starting point is [50,50]:

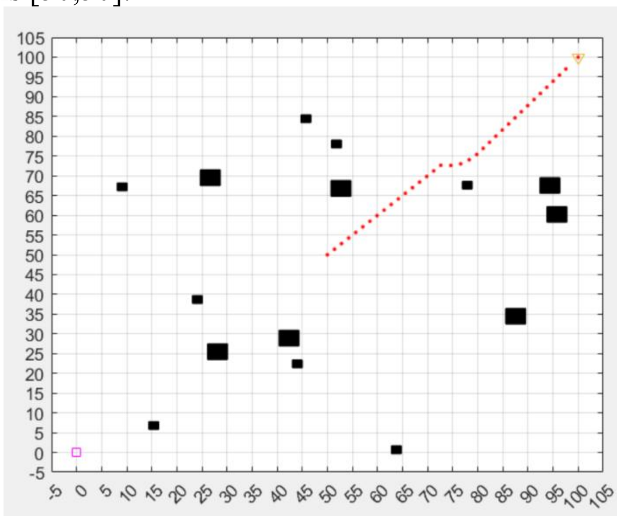

Figure 18. Results of path planning where the starting point is (50,50) and the target point is (100,100)

Change both x, y coordinates, the target point is [50,50]:

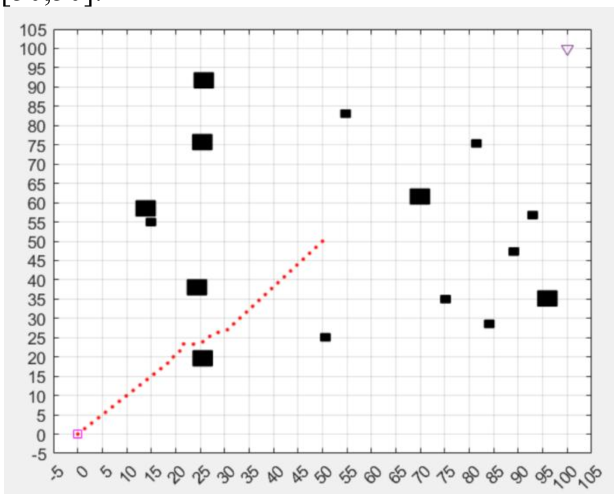

Figure 19. Results of path planning where the starting point is (0,0) and the target point is (50,50)

The first image shows that the autonomous vehicle starts from starting point [50,50] and reaches the target point [100,100]. The second image shows that the autonomous vehicle starts from starting point [0,0] and reaches the target point [50,50]. When the autonomous vehicle gets influenced by the obstacles, it redirects its route to bypass the obstacles. Though their starting points and target points are different, the distance between the two points are the same. That`s the reason why their driven routes are very similar.

Improvements to the Performance of Artificial Potential Field Method

This research demonstrates only the simplest implementation of the artificial potential field method. When applying this method in real life, there are often a number of problems that require corresponding improvements. The following sections will describe briefly several common problems and their solutions.

Improvement of the gravitational potential function: The existing gravitational potential function takes values that are proportional to d2(q, q_{goal}). In such a design, if the current point is too far away from the target point, a large gravitational potential is created, making the robot move too fast. This can be solved by using a segmented gravitational potential function, i.e. by reducing the power of d(q, q_{goal}).

Improvement of the repulsive potential function: The existing repulsive potential function takes values that depend on the distance between the current point and the nearest obstacle. In such a design, if the current point is equidistant from two obstacles, it may cause the robot to jump back and forth on the mid-line between the obstacles. In response, we can redefine the repulsive potential function as the distance between the current point and the nearest obstacle point. That is, regardless of the coordinate position of the obstacle itself, as long as a point on the obstacle is closest to the current point, this distance defines the repulsive potential function.

Improvement of calculation of distance: In general, we use linear distance to measure the distance between two points. But for pixel or grid maps, how can the distance between two pixel points or grids be calculated quickly and reasonably well? Reference gives a Brushfire algorithm to use.

The concern of the local minima problem: As shown in the figure, the artificial potential field method sometimes encounters the local minima problem. At the local minima point, although the gradient is zero, it is not the endpoint we want. For such cases, we generally introduce the idea of sampling planning, adding a perturbation (random walk) or backtracking at the local minima with a view to jumping out of the local minima. A method called wave-front planning can also be used to eliminate the problem of local minima by introducing a time parameter.

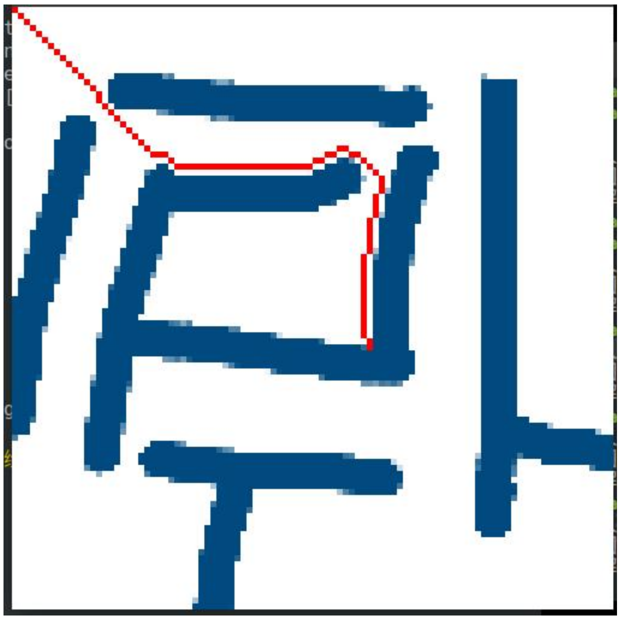

Figure 20. The artificial potential field method with the local minima problem

Conclusion

The algorithm of Artificial Intelligence (AI) continues developing and maturing, making automation of vehicles a possible thing. The advantage of artificial intelligence (AI) over humans is its great ability of real-control in a dynamic environment, allowing it to make accurate judgments about the environment in a very short time. Thus accidents can be avoided and the possibility of distracted driving can be eliminated.

For the realization of automation of vehicles, we raise a method called the artificial potential field method. The results tested on MATLAB R2021b show that the artificial potential field method is feasible for the path planning of the autonomous vehicle. Path planning refers to the fact that the robot finds a collision-free optimal path from the starting point to the target point in the workspace with obstacles, on the premise of the one or more optimization criteria among the shortest moving path, the shortest moving time, and the minimum working cost. Artificial potential field method involves constructing an artificial potential field to mimic the mechanism of action of the electric potential field and gravitational potential field when the starting point, target point and obstacle locations are known. This method can be widely used for the path planning of autonomous vehicle due to its efficient mathematical analysis and simplicity. However, when applying this method in real life, there are often a number of problems that require corresponding improvements, including the improvements of attractive and repulsive potential, calculation of distance and concern of local minima problem.

Conflict of Interests: the author has claimed that no conflict of interests exists.

References

1. Divakarla, K. P. , Emadi, A. , Razavi, S. , Habibi, S. , & Yan, F. . (2019). A review of the autonomous vehicle technology landscape. International Journal of Electric and Hybrid Vehicles, 11(4), 320.
2. Keshav, B. . (2015). Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology. IEEE.
3. Kato, S., Takeuchi, E., Ishiguro, Y., Ninomiya, Y., Takeda, K. and Hamada, T. (2015) 'An Open Approach to Autonomous Vehicles', IEEE Micro, Vol. 35, No. 6, November–December, pp.60–68.
4. Hong, T.H., Abrams, M., Chang, T. and Shneier. M. (2000) 'An intelligent world model for autonomous off-road driving', Computer Vision and Image Understanding Journal, Intelligent Systems Division, National Institute of Standards and Technology, 100 Bureau Drive Stop 8230, Gaithersburg, MD 20899−8230.
5. Patchett, C. (2015) A common systems architecture for autonomous vehicles', Presented at Advanced Engineering, November [Online], Available: http://www.easyfairs.com/fileadmin/groups/8/Advanced_Engineering/Advanced_Engineering_2016/AE_2015_presentations/Wednesday_-_Forum_5_-_13.30-14.00_-_Charles_Patchett_-_V2.pdf
6. Magnusson, M., Lilienthal, A. and Duckett, T. (2007) 'Scan registration for autonomous mining vehicles using 3DNDT', Journal of Field Robotics, Vol. 24, No. 10, May, pp.803–827.
7. Felzenszwalb, P.R., Girshick, D., McAllester, Ramanan, D. (2010) 'Object detection with discriminatively trained part-based models', IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, No. 9, September, pp.1627–1645.
8. Gruyer, D. , Magnier, V. , Hamdi, K. , Claussmann, L. , Orfila, O. , & Rakotonirainy, A. . (2017). Perception, information processing and modeling: critical stages for autonomous driving applications. Annual Reviews in Control, 44, 323-341.
9. McNaughton, N., Urmson, C., Dolan, J.M. and Lee, J.W. (2011) 'Motion planning for autonomous driving with a conformal spatio temporal lattice', Proc. 2011 IEEE International Conference on Robotics and Automation, May, Shanghai, China.
10. Fayyad, J. , Jaradat, M. A. , Gruyer, D. , & Najjaran, H. . (2020). Deep learning sensor fusion for autonomous vehicle perception and localization: a review. Sensors, 20(15), 4220.
11. Coulter, R. (1992) Implementation of the Pure Pursuit Path Tracking Algorithm, Robotics Institute, Tech. Report CMURI-TR-92-01.

12. SAE (2014) Taxonomy, and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems, SAE J3016_201401, Society of Automotive Engineers (SAE), January.
13. Li, Z. , & Wei, L. . (2015). Adaptive Artificial Potential Field Approach for Obstacle Avoidance Path Planning. 2014 Seventh International Symposium on Computational Intelligence and Design. IEEE.
14. Fan, X. , Y Guo, Liu, H. , Wei, B. , & Lyu, W. . (2020). Improved artificial potential field method applied for auv path planning. Mathematical Problems in Engineering, 2020(1), 1-21.
15. Choset. (2005). Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press.