Proceedings of the 37th Chinese Control Conference
July 25-27, 2018, Wuhan, China

Local path planning of autonomous vehicles based on A* algorithm with equal-step sampling

WANG Yijing, LIU Zhengxuan, ZUO Zhiqiang, LI Zheng

Tianjin Key Laboratory of Process Measurement and Control, School of Electrical and Information Engineering, Tianjin University,
Tianjin, 300072, P. R. China
E-mail: {yjwang, liuzhengxuan, zqzuo, aeolson960312}@tju.edu.cn

Abstract: Path planning is an important component in the study of autonomous vehicles. And local path planning is a pivotal issue, which will greatly influence the safety and comfort of the vehicle. On the premise of security, this paper proposes a novel A* algorithm with equal-step sampling based on vehicle kinematics model to improve the comfort of the path as much as possible. The algorithm is based on vehicle kinematics and introduces an enhanced cost function. The simulation results illustrate the feasibility of this approach on lane-keep and obstacle avoidance in different scenarios. In addition, the real vehicle test results of ”Smart Driving” autonomous vehicle of Tianjin University demonstrate the capability of real-time implementation.

Key Words: Autonomous vehicles, Local path planning, Vehicle kinematics model, Equal-step sampling

1 Introduction

With the development of society, cars have become indispensable in human life as one of the higher quality of life. However, traffic congestion and accident rate are increasing rapidly all over the world because of the increasing usage of motor vehicles and the lack of safety consciousness of drivers. Thus, the task of building intelligent transportation system (ITS) has become more urgent. Autonomous vehicles, as an essential part of ITS, have drawn considerable attention of experts and scholars in recent years. Autonomous vehicles incorporate environment perception and self-dependent location, decision-making and planning, motion control and other features to replace eyes, brain and hands of the driver [1]. An autonomous vehicle has the advantages of rapid response and safe traveling. Since the 20th century, the self-driving technology is rapidly expanding in America and Europe which is promoted by universities, related research institutes and some automotive companies. In 2009, Google has completed several prototype of autonomous vehicles and finished a road test of nearly 1 million kilometers [2]. Then in 2015, Google’s self-driving car completed the road test in California. In 2010, the autonomous vehicle which was made by VisLab laboratory at University of Parma in Italy passed through Moscow and Siberia, and finally arrived in Shanghai. The whole journey was nearly 8000 miles [3]. In 2011, Germany’s Free University of Berlin successfully completed a number of projects such as traffic flow, traffic lights, and roundabout traffic, etc. In 2012, three unmanned Volvo cars finished the road test at a speed of 90 km/h under the guidance of a truck on the road in Barcelona, Spain. In 2013, Mercedes-Benz completed an automatic driving road test with the length of 100 kilometers [4].

In order to enhance the security and comfort of cars, current automobile is no longer of pure mechanical structure. Many advanced scientific research results have been incorporated to improve the existing functions of cars. To make driving safer and more comfortable, automotive companies have developed Advanced Driver Assistance Systems (ADAS) such as Adaptive Cruise Control (ACC) and Lane Keeping Aid (LKA) [5]-[6]. The emergence of these technologies not only facilitates the operation of drivers, but also lays the foundation for path planning of autonomous vehicles.

Path planning arithmetic of autonomous vehicle mainly inherits the algorithms in robotics, such as A* algorithm [7]-[9], Rapidly-exploring Random Trees (RRT) algorithm [10]-[12], and potential field method [13], etc. More details about the development of path planning algorithms for autonomous vehicles can be found in [14]. A* algorithm is the most effective method searching the shortest path in static network. This algorithm is widely used in many kinds of robots because of its robustness and rapid response to environmental information. A* algorithm is an extension of Dijkstra algorithm. Compared with the original Dijkstra, it has several advantages such as fast convergence, apparent directional and small search space, etc. However, because of

This research was supported by the National Natural Science Foundation of China no. 61773281, no. 61673292, Tianjin Municipal Science and Technology Project, China, under Grant 17ZXRGGX00140.

the reliance of the search on the center of lattice cell and the combination of the path lengths in cost function, the path planned by the traditional A* algorithm is not smooth enough to meet the limitation of traffic lane line on the road with large curvature. As early as 2007, the autonomous vehicle ”Junior” which is made by Stanford University used an improved A* algorithm and won the second prize in the DARPA Urban Challenge organized by the U.S. Government [8]. Some research institutes in China also resort to A* algorithm on path planning of autonomous vehicle. For example, an improved A* algorithm through searching consecutive neighbourhoods which is generated on space-time occupancy grid map has been presented in [9]. RRT algorithm is widely used in the dynamic environment with high dimensional state and dynamic constraints because of its incremental growth characteristics. A RRT-based motion planner was developed by ”Talos” which was made by MIT team, and it successfully completed the competition in 2007 DARPA Urban Challenge [10]. In [11], the RRT-based motion planner has been implemented in the autonomous vehicle platform which called Intelligent Pioneer II.

Inspired by RRT algorithm, this paper proposes the idea of equal-step sampling to the strategy of local path planning, and puts forward a new type of cost function. Section 2 introduces the vehicle kinematic model. The detailed procedures of the algorithm are explained in Section 3. The simulation results in different scenarios are illustrated in Section 4. The algorithm has been successfully applied to the ”Smart Driving” autonomous vehicle of Tianjin University. The test results on real vehicle are described in Section 5.

2 Vehicle kinematic model

The kinematic equation describes the motion pattern on the basis of geometry (e.g. position, velocity, acceleration). In reality, a reasonable choice of vehicle model is very important to the complexity and performance of planning and control modules in unmanned system. If the kinematic and dynamic constraints are considered in path planning, the tracking performance will become much better. However, as the dynamics of vehicle movement on the ground is very complicated, we need use several state variables to model a complex system with differential equation. As a matter of fact, the complexity imposed on motion model usually cannot lead to the improvement of accuracy. In contrast, it may cause to the decrease of real-time characteristics. Therefore, simplification of the motion model is indispensable. In such a case, we can use the vehicle kinematic model to make the results feasible under the geometric constraints in path planning module [15].

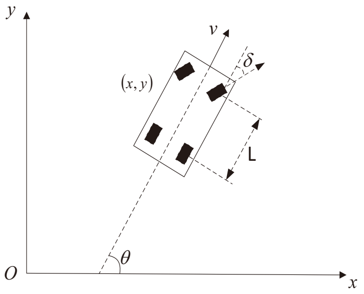

Fig. 1: Vehicle kinematic model.

Vehicle kinematic model is depicted in Fig. 1. The relationship between the front-wheel steering angle and the steering radius can be described as

$$\tan\delta = \frac{L}{R} \tag{1}$$

where L and R denote the wheelbase and the steering radius of the vehicle, and δ is the front-wheel steering angle. Based on it, the vehicle kinematic model with two degrees of freedom is formulated as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\delta}{L} \end{bmatrix} v \tag{2}$$

where the vector $[x, y, \theta]^T$ denotes the pose (position and orientation) of the vehicle in the global coordinate system. The variable v denotes the velocity of the vehicle. $[v, \delta]^T$ are the control variables in this model accordingly.

3 A* algorithm with equal-step sampling

A* algorithm is based on a grid map that represents environmental information, which belongs to graph search method. The traditional A* algorithm was presented in [16]-[17]. Graph search methods are often used in path planning on robots. The basic idea can be described as finding a collision-free path that connects the starting point with the goal point satisfying system constraints when the representation of robot and the location of obstacles is known [18].

In general, the requirements of autonomous vehicle path planning are comfortable, real-time and reliable. As mentioned in Section 2, the tradeoff between model completeness and real-time constraints has to be addressed. The accurate algorithm usually involves heavy computation burden and discontents the real-time requirements [19]. According to the above requirements on autonomous vehicle, we propose an equal-step sampling method based on vehicle kinematic model to the local path planning strategy and put forward a new type of cost function. In addition, in order to take into account the size of the vehicle, we use several circles to represent it, which is called secure search domain. The specific implementation process is as follows.

Step 1) Determine the search step length in each sampling. In our algorithm, the search step length is chosen as

$$Step = \frac{vT}{l} \tag{3}$$

where $Step$ is the search step length, and l, v and T represent the precision of the grid, the velocity of the vehicle and the updating time of local path, respectively.

Step 2) Calculate the coordinates of sampling points using the model in (2). The coordinate of each search node (x_{search}, y_{search}) in vehicle body coordinate system can be formulated as

$$\begin{gathered} x_{search} = x_{parent} + Step \times \cos(\theta_{parent} + i \times \varphi), \\ y_{search} = y_{parent} + Step \times \sin(\theta_{parent} + i \times \varphi), \\ i \in \{-N, -N+1, \ldots, 0, \ldots, N-1, N\} \end{gathered} \tag{4}$$

where (x_{parent}, y_{parent}) is the coordinate of the parent node, and θ_{parent} describes the heading angle. At the beginning, the starting node is the parent node and the heading angle is zero. The number of chosen nodes at each sampling is decided by N. That is, the adjacent nodes are chosen with fixed angular difference $\varphi = \frac{\phi_{max}}{N}$, where $\phi_{max} = \frac{\tan \delta_{max}}{L} vT$ is the limitation of the change of heading angle at each sampling moment. i is the count of the node, which determines the motion direction. When i is positive, the vehicle turns left, otherwise it turns right. The heading angle of the adjacent node is $\theta_{search} = \theta_{parent} + i \times \varphi$. The vehicle body coordinate system is displayed in Fig. 2.

Step 3) Determine whether the area around current search node is safe or not. For the sake of safety, the circular secure search domain is used in our work due to its rotational symmetry. The radius of each circle is

$$r_{safe} = \max\left(\frac{Step}{2}, \frac{L_{vehicle}}{2}\right) \tag{5}$$

where $L_{vehicle}$ is the length of the vehicle. For each search node, we assess two circular domains. One is related to this node, and the other is the midpoint of this node and its father node. If the area covered by these two circular domain is obstacle-free, the current search node is safe. Otherwise it is dangerous and the corresponding cost function is infinite. After that, the safe search nodes can be added into the open list.

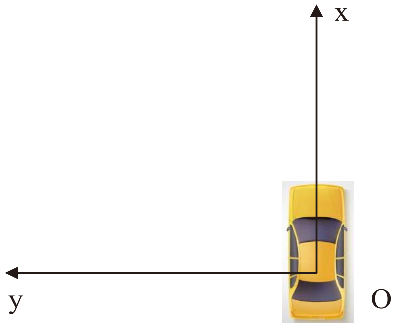

Fig. 2: Vehicular coordinate system.

Step 4) Choose the node with lowest cost as current search node (i.e. next parent node) at each sampling. For the traditional A* algorithm, the cost function is often described as in [20]

$$f(n) = g(n) + h(n) \tag{6}$$

where $g(n)$ and $h(n)$ represent the cumulative and heuristic cost. In this paper, an enhanced cost function with more information is defined as

$$f(n) = K_1 g(n) + K_2 h(n) + K_3 p(n) \tag{7}$$

$$\begin{cases} g(n) = g_1 L_{acc}(n) + g_2 D_{acc}(n) \\ h(n) = h_1 L_{est}(n) + h_2 D_{est}(n) \\ \quad p(n) = (\frac{\theta(n) - \theta(n-1)}{\varphi})^{\alpha_1} \end{cases} \tag{8}$$

where K_1, K_2 and K_3 are positive scalar weights. $g(n)$ and $h(n)$ are the same as in (6). We add $p(n)$ as the penalty term in each sampling. In more detail, $L_{acc}(n) = 1 + L_{acc}(n-1)$ and $D_{acc}(n) = \frac{\theta_{cur} - \theta_{father}}{\varphi} + D_{acc}(n-1)$ describe the cumulative step length and steering cost from the starting node to the current node. Similarly, $L_{est}(n) = \frac{|x_{cur} - x_{goal}| + |y_{cur} - y_{goal}|}{Step}$ and $D_{est}(n) = \frac{|\arctan((y_{goal} - y_{cur})/(x_{goal} - x_{cur}))| - \theta(n)}{\varphi}$ formulate the ideal projected step length and steering cost from the current node to the goal node. In order to reduce the computational complexity, the Manhattan Distance is adopted in $L_{est}(n)$. $p(n)$ is defined as the fixed cost from the father node to the current node. $p(n)$ only uses the turning cost, since step length cost is the same in this function. All scalar weights are positive in the cost function. As mentioned above, at the starting node, the cumulative cost is $g(0) = 0$, and the penalty term is $p(0) = 0$. The initial cost consequently can be computed by plugging these into (7)

$$f(0) = K_2(h_1 L_{est}(0) + h_2 D_{est}(0)). \tag{9}$$

Step 5) Repeat the above steps until the following conditions are satisfied

$$\begin{cases} L_{est}(n) \leq 2, \\ D_{est}(n) \leq 1. \end{cases} \tag{10}$$

In this case, we can see that the current search node is approaching to the goal node. Then the closed list will be regarded as the feasible path. If the open list is null in the searching process, there is no feasible path at present.

A pseudo code of the overall algorithm for our algorithm is shown below.

Algorithm 1 A* algorithm with equal-step sampling

```
$open, closed \leftarrow \emptyset$
$start.[f, Step, direction, parent, child], n \leftarrow \emptyset$
$open.push(start)$
$closed.push(start)$
while $open \neq \emptyset$ do
    if $curr.child \neq \emptyset$ then
        $curr \leftarrow curr.childwith$ $\min(f)$
    else
        $closed.delete(curr)$
        $curr \leftarrow parent$
    $closed.push(curr)$
    $open.delete(curr)$
    for each $a \in adjacent$ do
        $next.state \leftarrow move(curr, a)$
        if $IsSafe(next.state) \wedge a \notin open$ then
            $child.push(a)$
            $open.push(a)$
            $g \leftarrow calculateCumulative(start \rightarrow a)$
            $h \leftarrow calculateHeuristic(a \rightarrow goal)$
            $p \leftarrow calculatePenalty(curr \rightarrow a)$
            $a.f \leftarrow g + h + p$
    $i \leftarrow i + 1$
    if $IsGoal(curr)$ then
        break
return $closed$
```

4 Simulation Results

In order to verify the feasibility and superiority of the proposed algorithm in this paper, we compare it with the traditional algorithm which selects the center of the lattice cell as the search node. The comparison results in obstacle avoidance behavior and on winding road are plotted in Fig. 3 and Fig. 4. Through comparison, the results of A* algorithm with equal-step sampling are smoother than the traditional one. As depicted in Fig. 3, the traditional algorithm is inflexible in swerve and that may increase several difficulties on tracking and control. In order to keep in the lane on winding road, the traditional algorithms need additional item of drifting of the center line of the lane in the cost function but still have some problems with swerve (as shown in Fig 4). However, the proposed algorithm in this paper can plan a smooth path for obstacle avoidance and lane keeping due to the existence of steering cost.

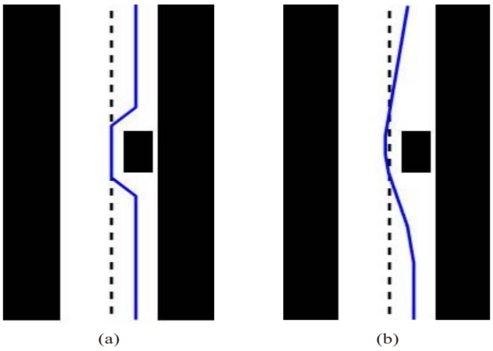

Fig. 3: Results comparison in obstacle avoidance behavior. (a) The A* algorithm with the searching node at the center of each lattice cell. (b) The path planning algorithm in this paper. The black dotted line is lane mark, and the blue solid line is the search path.

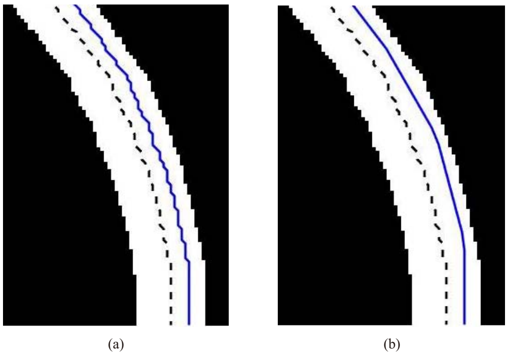

Fig. 4: Results comparison on winding road. (a) The A* algorithm with the searching node at the center of each lattice cell. (b) The path planning algorithm in this paper.

Simulation results in different scenarios are illustrated by Fig. 5. The general design parameters are given in Table 1. The size of local grid map is $25 \times 25[m^2]$.

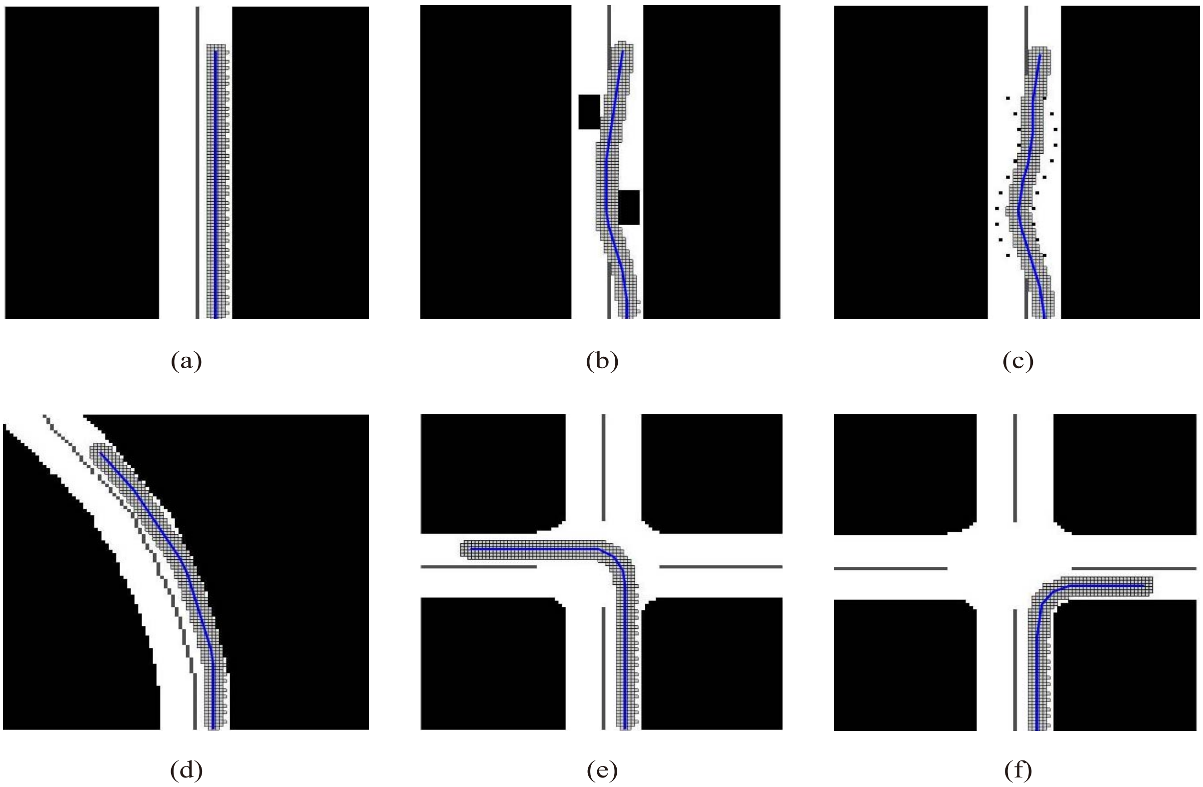

Fig. 5: Plan results in different settings. (a) straight road, (b) obstacle avoidance on straight road, (c) S-bend passage, (d) winding road, (e) left-turn at the intersection, (f) right-turn at the intersection. The gray area is the secure search domain.

Table 1: General Design Parameters For MATLAB Simulation

$v_{des} = 12.5[m/s]$	$T = 0.1[s]$	$l = 0.25[m]$
$N = 4$	$K_1 = 0.6$	$K_2 = 1.6$
$K_3 = 0.5$	$g_1 = 1$	$g_2 = 1.2$
$h_1 = 1$	$h_2 = 0.6$	$\alpha_2 = 1.2$

5 Experimental Results

The algorithm proposed in this paper has been implemented in the "Smart Driving" autonomous vehicle platform in Tianjin University in Fig. 6. The vehicle is equipped with a LIDAR, three MMW radars, two cameras and a GPS/INS receiver. The general design parameters are given in Table 2. In our work, the front-wheel steering angle δ is restricted in $[-30°, 30°]$. The wheelbase is $L_{vehicle} = 2.72m$. The size of local grid map is $20 \times 30[m^2]$. The planning time of each grid is $30 \sim 60\ ms$. On straight roadway, the planning time is around $30 \sim 40\ ms$. Fig. 7 is the images of the real vehicle test. The black scattered points are the obstacles detected by the LIDAR and the red part is the route planned by this algorithm.

6 Conclusion

The traditional A* algorithm and some improved A* algorithms have some common problems, such as asymmet-

Fig. 6: "Smart Driving" autonomous vehicle of Tianjin University.

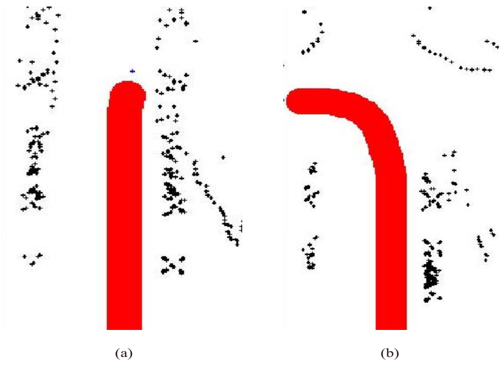

Fig. 7: The real-time results.

Table 2: General Design Parameters For Vehicle Test

$v_{des} = 10[m/s]$	$T = 0.1[s]$	$l = 0.125[m]$
$N = 4$	$K_1 = 0.8$	$K_2 = 1.52$
$K_3 = 0.25$	$g_1 = 1$	$g_2 = 1.2$
$h_1 = 1$	$h_2 = 0.6$	$\alpha_2 = 1.2$

ric distance between two adjacent nodes and no limitation on search direction. These problems may lead to some preprocess before tracking and motion control: node-deletion and curve fitting, for instance. In this paper, we have proposed a novel A* algorithm with equal-step sampling. The method of selecting search node based on equal-step sampling limits the direction of movement. The enhanced cost function not only includes distance cost, but also considers steering cost, and penalizes each step by using the cost on every movement. The above factors can avoid a sudden turn as far as possible. In this way, this path planning strategy satisfies the constraint of steering and improves smoothness of the path. In order to reduce the computational complexity, the Manhattan Distance is adopted in the cost function. The circular secure search domain is more suitable to the proposed algorithm in this paper due to its rotational symmetry. Meanwhile, the secure search domain ensures that it never makes the appearance that the connecting line between each two adjacent nodes across obstacles. In conclusion, the proposed algorithm in this paper meets the basic three requirements on path planning.

Several simulation and experimental tests have been done to verify the feasibility of the proposed algorithm, which can be well applied to simple scenarios.

References

[1] Alia C, Gilles T, Reine T, et al. Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method, *Intelligent Vehicles Symposium. IEEE*, 674-679, 2015.

[2] Markoff J. Google cars drive themselves, in traffic, *New York Times*, 2010.

[3] Bertozzi M, Bombini L, Broggi A, et al. The VisLab intercontinental autonomous challenge: 13,000 km, 3 months, ... no driver, *Its World Congress*, 2010.

[4] Ziegler J, Bender P, Schreiber M, et al. Making Bertha drive-An autonomous journey on a historic route, *IEEE Intelligent Transportation Systems Magazine*, 6(2):8-20, 2014.

[5] Neale V L, Dingus T A, Klauer S G, et al. An overview of the 100-car naturalistic study and findings, *International Technical Conference on the Enhanced Safety of Vehicles*, 787, 2005.

[6] Vaa T, Penttinen M, Spyropoulou I. Intelligent transport systems and effects on road traffic accidents: state of the art, *IET Intelligent Transport Systems*, 1(2):81-88, 2007.

[7] Yoon S, Yoon S E, Lee U, et al. Recursive path planning using reduced states for car-like vehicles on grid maps, *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2797-2813, 2015.

[8] Montemerlo M, Becker J, Bhat S, et al. Junior: The Stanford entry in the urban challenge, *Journal of Field Robotics*, 25(9): 569C-597, 2008.

[9] Xin Y, Liang H, Mei T, et al. A new occupancy grid of the dynamic environment for autonomous vehicles, *Intelligent Vehicles Symposium Proceedings*, 787–792, 2014.

[10] Kuwata Y, Teo J, Karaman S, et al. Motion planning in complex environments using closed-loop prediction, *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2008.

[11] Du M, Chen J, Zhao P, et al. An improved RRT-based motion planner for autonomous vehicle in cluttered environments, *IEEE International Conference on Robotics and Automation. IEEE*, 4674-4679, 2014.

[12] Moses E D B , Anitha G. Goal directed approach to autonomous motion planning for unmanned vehicles, *Defence Science Journal*, 67.1: 45, 2017.

[13] Ji J, Khajepour A, Melek W W, et al. Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints, *IEEE Transactions on Vehicular Technology*, 66(2):952-964, 2017.

[14] Paden B, Cap M, Yong S Z, et al. A survey of motion planning and control techniques for self-driving urban vehicles, *IEEE Transactions on Intelligent Vehicles*, 1(1):33-55, 2016.

[15] Ailon A, Berman N, Arogeti S. On controllability and trajectory tracking of a kinematic vehicle model, *Automatica*, 41(5):889-896, 2005.

[16] Borenstein J, Koren Y. Histogramic in-motion mapping for mobile robot obstacle avoidance, *IEEE Transactions on Robotics and Automation*, 7(4): 535-539, 1991.

[17] Konolige K. Improved occupancy grids for map building, *Autonomous Robots*, 4(4): 351-367, 1997.

[18] Lavalle S. Planning algorithms, *Cambridge University Press*, 2006.

[19] Reif J, Wang H. The complexity of the two dimensional curvature-constrained shortest-path problem, *The Workshop on the Algorithmic Foundations of Robotics on Robotics: the Algorithmic Perspective: the Algorithmic Perspective*, 49-57, 1998.

[20] Esmaeili E, Azizi V, Samizadeh S, et al. A new exploration method based on multi-layer evidence grid map (MLEGM) and improved A* algorithm for mobile robots, *IEEE, International Conference on TOOLS with Artificial Intelligence*, 953-957, 2012.