PAPER

# A Practical and Optimal Path Planning for Autonomous Parking Using Fast Marching Algorithm and Support Vector Machine

Quoc Huy DO[†a)], *Nonmember*, Seiichi MITA[†], *Member*, *and* Keisuke YONEDA[†], *Nonmember*

**SUMMARY** This paper proposes a novel practical path planning framework for autonomous parking in cluttered environments with narrow passages. The proposed global path planning method is based on an improved Fast Marching algorithm to generate a path while considering the moving forward and backward maneuver. In addition, the Support Vector Machine is utilized to provide the maximum clearance from obstacles considering the vehicle dynamics to provide a safe and feasible path. The algorithm considers the most critical points in the map and the complexity of the algorithm is not affected by the shape of the obstacles. We also propose an autonomous parking scheme for different parking situation. The method is implemented on autonomous vehicle platform and validated in the real environment with narrow passages.

***key words:*** *path planning, autonomous parking, fast marching, support vector machine*

## 1. Introduction

Recently, several automotive manufacturers have introduced the concept of a "self-parking car" whereby a car can be maneuvered into a parking spot under automated control. An autonomous parking assistance system generally performs complicated tasks including environment mapping, path planning, and path tracking. Path tracking is composed of two components: a local path planner which generates a short range (a local path) to follow a globally planned path, and a lower level motion command (e.g. accelerating, steering, braking and so on) controller. On account of wide demand for autonomous parking, many parking path planning algorithms have been developed. The path planning for autonomous parking, mainly operated in a semi-structured environment is different from the normal on-road path planner which mainly runs in the structured environment. In [1], Pradalier designed a parking assistance system with the help of a database for storing the pre-calculated control profile parameters and the resulting movement. However, it is too heavy for an on-board system to maintain such a database or the resolution is limited considering the computing resource constraints. In [2], Paromtchik presented an iterative algorithm for parking maneuver based on ultra-sonic range data processing and sinusoidal function to control the steering angle and longitudinal velocity of the vehicle. A velocity and steering command generation method for autonomous parking was presented by same author in [3]. Oetiker [4] presented a low-cost and memory efficient algorithm based on navigation-field for a semi-autonomous vehicle parking assistant. Taix [5] presented a multi-level solution which separates the planning task into holonomic and non–holonomic problems and the main idea was adapted to the vehicle parking problem by Muller in [6]. Such paths are feasible to vehicles with limited steering speed and allow for parking maneuvers at continuous longitudinal motion. In [7], M.F. Hsieh proposed an arcs-based parking path planning method which was used by OSU-ACT team in the DARPA Urban Challenge 2007. However, these above mentioned methods perform autonomous navigation only in a short range and are not able to plan complex navigation tasks through entire parking structures. K. Kondak [8] applied numerical methods and artificial potential field to solve the nonlinear optimization for autonomous parking but the algorithm is very time consuming. In [9], M. Wada presented a multilevel driver assistance system based on path planning and human interface to assist parking. A continuous curvature trajectory design and feed-forward control for parking method based on two-step path planning was proposed by B. Muller [10]. Dubins [11] presented an algorithm for computing a shortest path between two postures in the plane for a vehicle having a limited curvature. Reed and Sheep [12] extent Dubin's work with the consideration of moving forward and backward maneuvers. The weakness of these methods is that the appearances of the obstacles are not considered while a path is being generated. Other planners [13]–[15] have been successfully dealing with vehicle's dynamic constraints by using probabilistic path planning methods. In [14] L. Han proposed a unified autonomous parking planner based on bi-directional RRT but this method does not consider the number of gear transition. Kuwata [15] developed algorithm of closed loop rapidly-exploring random tree (CL-RRT) for the path planning task running in real time. The CL-RRT is able to plan motions on different scenarios. Nevertheless, these methods does not provide optimal result. In [16], Likhachev presented a long-range path planning algorithm based on Anytime Dynamic Astar to efficiently generate complex plans overlarge, obstacle-laden environments. However, these methods have limitations and do not produce optimized path to the goal position. In Japan, generally parking areas are cluttered environment with narrow passages which makes the autonomous parking more complicated. For cluttered and narrow passages we always need the safest path, that is, the path that can provide maximum distance from the

Manuscript received May 13, 2013.
Manuscript revised August 23, 2013.

[†]The authors are with Toyota Technological Institute, Nagoya-shi, 468–0034 Japan.
a) E-mail: sd09601@toyota-ti.ac.jp

DOI: 10.1587/transinf.E96.D.2795

obstacles around it.

It is necessary to overcome the above mentioned problems including computational costs and localization uncertainty, so we proposed a new method to generate parking path with maximum-clearance path planning: We proposed a safe and smooth path planning method based on the binary classification Support Vector Machine (SVM) method. The method discussed in this paper is an implementation of our previous works [17], [18] to apply in the real environment with
(1) Forward and backward maneuver are considered to satisfy the final position constraint and the number of gear change are also considered.
(2) A modified Fast Marching Method (FMM) which is applicable for the parking path planning problem and provide the optimal path.
(3) Parking schemes for the vehicle to handle different parking situations.

The structure of this paper is as follow: Sect. 2 describes the overview of an autonomous parking assistant system and the schemes for autonomous parking. Section 3 explains the global path planning method. Experimental results are presented in Sect. 4. Finally, the conclusion is presented in Sect. 5.

## 2. Proposal Autonomous Parking Scheme

### 2.1 Autonomous Path Planning System Overview

The whole system's architecture and data flow diagram of autonomous vehicle components are shown in Fig. 1. The vehicle sensors data including laser data and IMU/odometer have been used for preparing the global SLAM map as an occupancy grid map. The global map is prepared in advance. We use the generated global map as a reference for vehicle path planning.

The autonomous path planning is divided into two phases such as global and local path planning.
(1) The global path planner will generate a path based on the known or recorded data (the occupancy grid map).
(2) Based on the generated global path, at each timestamp of the vehicle control, we generate the local path to avoid the dynamic obstacles or the newly appeared obstacles.
In a longer timestamp, the global path can be called to generate a new global path. The details of the local planners are presented in our previous work [17], [18].

### 2.2 Proposed Autonomous Pparking Scheme

We use the planners proposed in Sect. 3 and [18] to address the realistic navigation scenario for an autonomous vehicle, where narrow passages, moving obstacles must be taken into consideration. When the vehicle begins to enter the area for parking, two following situations can be exist.
A. There is no specific parking position, the vehicle have to search for it. The parking scheme is shown in Fig. 2.
Step 1: Generate a path from current position or parking area's entrance to the exit
Step 2: Follow the generated path and detect parking location.
Step 3: If one or more parking locations are detected, the

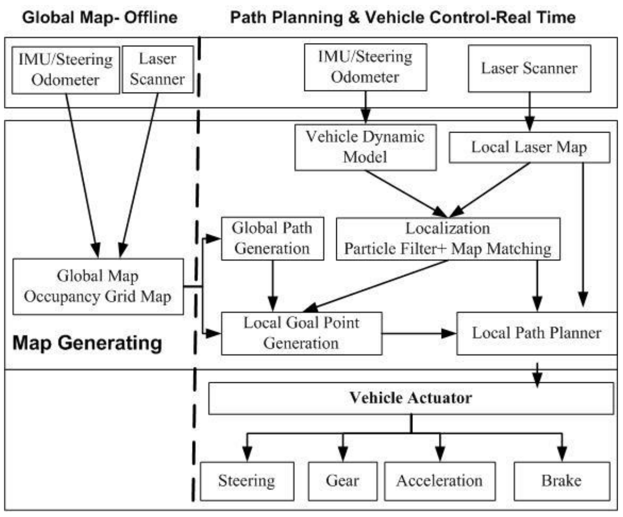


**Fig. 1** Autonomous parking architecture and data flow.

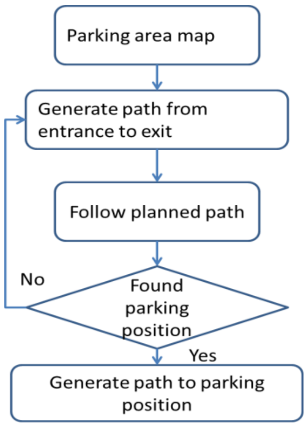

Parking area map
Generate path from entrance to exit
Follow planned path
No
Found parking position
Yes
Generate path to parking position

**Fig. 2** Autoparking scheme.

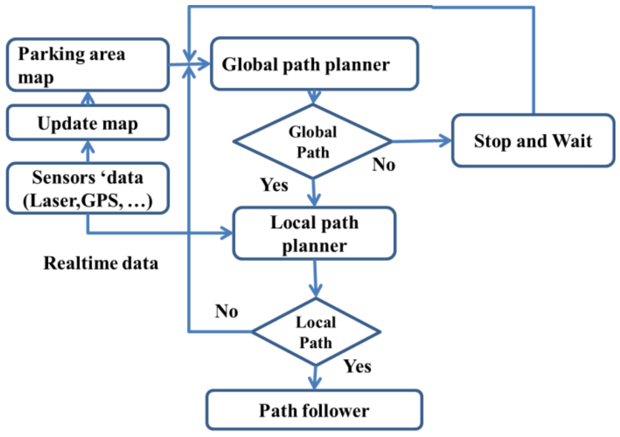
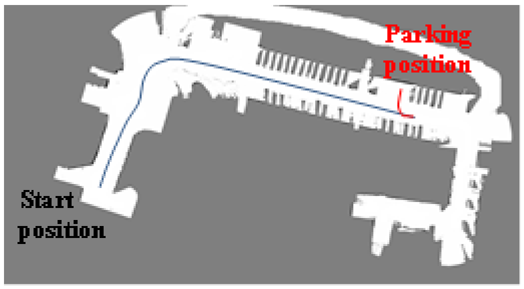
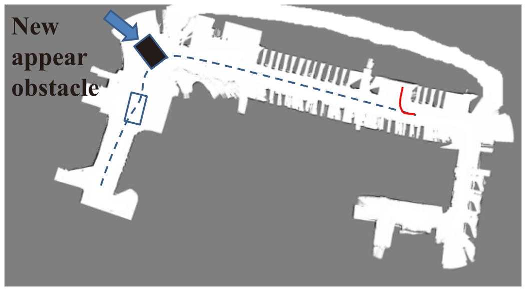


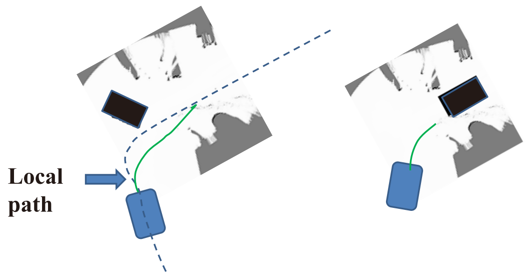
**Fig. 3** Parking with known position scheme.



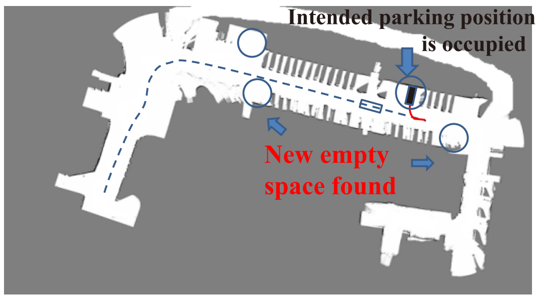
a) Planned global path



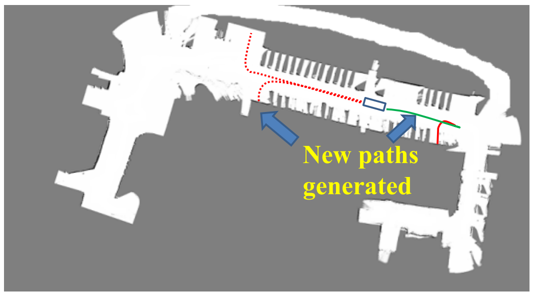
b) New obstacle interfere the planned path



c) Local path planner reacts.



d) Change determined parking position



e) New candidate path for parking.

**Fig. 4** Example of a parking situation and parking scheme.

global path planner will be called to generate the path to these locations. The final parking location will be chosen based on the cost of distance travel, safety distance and number of backward maneuver. The parking position can be selected to minimize the cost function (1).

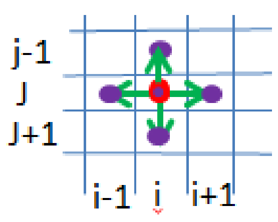
$$C(path) = w_1\, \text{length}(path) + w_2\, \text{margin}(path) + w_3\, \text{number\_of\_gear\_change}(path) \quad (1)$$

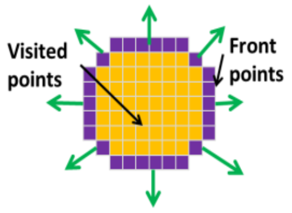
Step 4: The rest of the tasks are similar to Scheme B.
Step 5: If the vehicle can reach to the parking position, the path will be stored in the library to be used in the future.
B. The parking position is already known in advance.
Step1: The global path planner will generate a path from current position to parking position.
Step 2: While the vehicle travelling, the local path planner will be called to generate local path to follow global waypoints.
Step3: If a moving obstacle appears, the local path planner will generate a local path to avoid it.
Step 4: if no local path exist, the vehicle will stop and call the global path planner to generate a new path (go to step 1).
Step 5: If no new path exist then the vehicle has to do the loop stop and wait, try to generate new path. If after an n number of attempts the vehicle still cannot find a new path then it has to go backward along the travelled path and generate a new path again. If a new path is generated then go to step 2.
Step 6: if the goal (pre-determined parking position) is occupied then the vehicle have to detect a new parking position.

Figure 4 shows an example of parking situation. Figure 4.a shows that in the beginning, a parking position is determined and a global path is generated from the vehicle current position to the parking position. As shown in Fig. 4.b, while the vehicle is following the generated global path, a new car (new obstacle) appears and makes the path unsafe. The local path planner will generate a new local path to avoid the obstacle as shown in Fig. 4.c. Figure 4.d shows that when the vehicle reached the planned parking position, it found out that the position was occupied. Thus, the vehicle has to find a new position to park. Based on the vehicle stored information in the map, several empty positions are selected and the vehicle will generate paths to these candidates as shown in Fig. 4.e. The parking position and path will be determined based on the cost function (1) and the vehicle will follow this new generated path.

## 3. Global Path Planning with Improvement of Fast Marching Method

The autonomous parking path planning must take into account a wide variety of factors such as the followings:
(1) The safety distance of the path to the obstacles (which include the static or priory known obstacles and the newly appear or moving obstacles).
(2) The total travel distance of the path.
(3) The length of path's segments that the vehicle has to





a) 2D neighbors b) Illustration of Front propagating

**Fig. 5** 2D FMM.

drive in backward maneuver.
(4) The number of times that the vehicle has to switch the gear from forward to backward and vice versa.
(5) The curvature of the path to satisfy the kinematic constraints.
We assume that the vehicle is equipped with sensors for both the front and the backward so that driving forward or backward has the same meaning for the autonomous vehicle.

The objective of the autonomous parking planner is to minimize the following cost defined over a path.

$$\mathrm{C}(\mathrm{p}_i) = w_1 \sum_{i=1}^{N} d(p_i) + w_2 \sum_{i=1}^{N-1} |g(p_i) - g(p_{i-1})| + w_3 \sum_{i=1}^{N} \Delta p_i + w_4 \sum_{i=1}^{N-1} |\kappa(p_i)| \qquad (2)$$

where $N$ is the number of path points, $p_i$ ($i$ = 1..N) are the points on the path, $d(p_i)$ is the reversed distance from the path point to the obstacles, the distance between path points $\Delta p_i = \|p_i - p_{i-1}\|$, $g(p_i) \in \{1, -1\}$ determine the motion gear: (1 = forward) and (−1 = backward); $\kappa(p_i)$ is the curvature of the path at point $p_i$ and $w_1$, $w_2$, $w_3$, $w_4$ are the weight factors.

The proposed global path planning is the improvement of the fast marching method to apply for parking path planning.

## 3.1 The Fast Marching Method

The Fast Marching Method (FMM) [19] has been introduced to solve the static Hamilton Jacobi (Eikonal) equation $|\nabla \mathrm{U}(x)| \cdot \mathrm{F}(x) = 1$. Here, $\mathrm{F}(x) > 0$ is the front moving speed at point $x$ and $\mathrm{U}(x)$ is the function of travelled time. $1/\mathrm{F}(x)$ can also be known as objective cost and $\mathrm{U}(x)$ is known as cost to go from start point to point $x$. As implemented in the robotic navigation, the formulation can be interpreted as path planning optimal problems (with the consideration of Eq. (2)). Sethian [20] proposed to use the Godunov Hamiltonian which is a one-sided derivative.

At each point (i, j) in the search graph (Fig. 5 a), the unknown cost value u satisfies:

$$f_{i,j}^2 = (\max\{\mathrm{u} - \mathrm{U}_{i-1,j}, \mathrm{u} - \mathrm{U}_{i+1,j}, 0\})^2 + (\max\{\mathrm{u} - \mathrm{U}_{i,j-1}, \mathrm{u} - \mathrm{U}_{i,j+1}, 0\})^2 \qquad (3)$$

where U is the known cost value of the neighbor node of $(i, j)$, $f_{i,j}$ is the cost to travel between the nodes in the graph. Starting with an initial position for the front, the method systematically marches the front outwards one grid point at a time as illustrated in Fig. 5 b.

The FMM algorithm is as follow:

```
1) Definitions
  • Visited is the set of all points at which the cost to go value U has been reached and will not be changed;
  • Front is the set of next points to be examined and for which, an estimate U of $u$ was computed using Eq. (3) only from Visited points;
  • Unvisited is the set of all other grid points, for which their cost to go value U is not yet estimated.
2) Initialization
  • The starting point $\mathrm{p}_0$ is put to Visited set, $\mathrm{U}(\mathrm{p}_0) = 0$;
  • Front - the initial front is narrowed to the neighbors of $\mathrm{p}_0$ with initial values $\mathrm{U}(\mathrm{p}) = \mathrm{f}(\mathrm{p})$;
  · Unvisited is the set of all other grid points, $\mathrm{U} = \infty$;
3) Marching Loop
  • Let $\mathrm{p} = (\mathrm{i}_{\min}, \mathrm{j}_{\min})$ be the Front point with the smallest cost U;
  • Move it from the Front set to the Visited set (i.e. $\mathrm{Ui}_{\min}, \mathrm{j}_{\min}$ is frozen);
  • For each neighbor (i, j) of $(\mathrm{i}_{\min}, \mathrm{j}_{\min})$:
    a) If (i, j) is Unvisited, add it to the Front set and compute a first estimate U of U using Eq. (3)
    b) If (i, j) is in Front, update the value $\mathrm{U}_{i,j}$ using Eq. (3).
```

An optimal path is generated by tracking backward the points from goal to start via gradient descent by using the computed values $u$.

The FMM has shown to generate a smoother path than A* in 2D environment [21], [22]. However, major of existing applications of FMM for path planning are in 2D environment and the result paths are not feasible enough for the car-like vehicle to follow [23]. In [24] Clement proposed a method for underwater vehicle path planning but this method does not consider either the position requirement (e.g. the heading angle at the start or goal point) or the backward maneuver.

## 3.2 Improvement of Fast Marching Method

The path planning is performed on a 4D continuous search space $(x, y, \theta, g)$, where (x, y) are 2D coordination, $\theta$ is the vehicle heading angle and the fourth dimension ($g \in \{1, -1\}$) represents the current driving maneuver (forward or reverse).

Instead of using directly FMM on this 4D space, in this paper, we propose an improvement of Fast Marching Method (FMM) for car-like vehicle path planning with:

(1) Increase the speed of the method by performing a preprocessing or off-line calculation step to reduce the search space.

(2) Add the obstacles' clearance to the path based on a safety field.

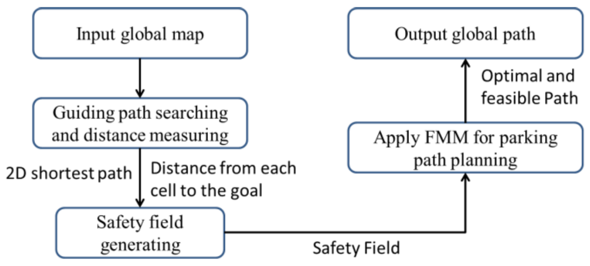

Input global map
Guiding path searching and distance measuring
2D shortest path
Distance from each cell to the goal
Safety field generating
Safety Field
Apply FMM for parking path planning
Optimal and feasible Path
Output global path

**Fig. 6** Global path planning.

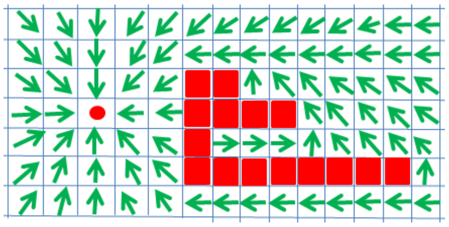


**Fig. 7** The travel time gradient at each point gives the estimation of travel distance back to the goal.

(3) A continuous search space with consideration of vehicle kinematic constraints and backward-forward maneuver.

The overview of global path planning algorithm is described in Fig. 6. Our method based on three main steps which are described in the following sections.

A. Guiding Path Searching and distance measuring

We search the shortest path from the goal point to the start position in 2D graph.

In this step, the cost value for travel between the grid cells is $f_{i,j} = 1$; and the neighbors are defined based on the 4 connectivity of the cell as shown in Fig. 5 a. Notice that only Visited points are considered to solve Eq. (3). We examine neighbors of point (i, j) in 4-connexity. Let $U_{A1} = \min(U_{i-1,j}, U_{i+1,j})$ and $U_{B1} = \min(U_{i,j-1}, U_{i,j+1})$. Assuming that $u \geq U_{B1} \geq U_{A1}$ Eq. (3) becomes:

$$f_{i,j}^2 = (u - U_{A1})^2 + (u - U_{B1})^2 \tag{4}$$

$$\text{Let } \Delta = 2f_{i,j}^2 - (U_{A1} - U_{B1})^2 \tag{5}$$

If $\Delta > 0$ then

$$u = \frac{U_{A1} + U_{B1} + \sqrt{\Delta}}{2} \tag{6}$$

$$\text{Else } u = U_{A1} + f_{i,j} \tag{7}$$

This step helps us avoid the local trap such as the U-shaped obstacles, dead-ends and acquires the distance from each cell on the graph to the goal as shown in Fig. 7. In other word, we can find path from different point to the goal point. This distance is called 2D-dist-to-goal and lately can be used to speed up the searching in 4D space.

B. SVM for Safety Field

We pick up all the points on the borders (left and right side) along this the guiding path as labeled data points to input as training data sets for SVM as shown in Fig. 8.

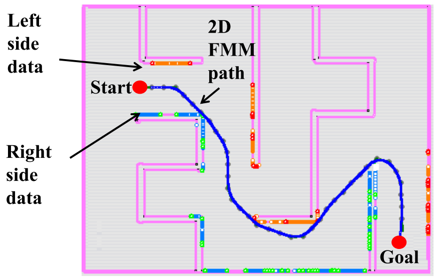


**Fig. 8** Generate two separate datasets of obstacles data points using RRT guided path.

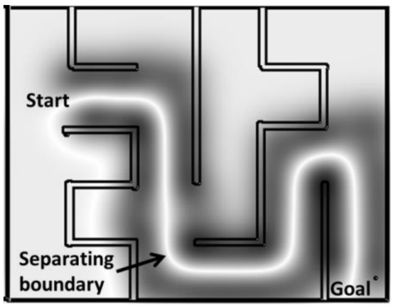


**Fig. 9** The virtual safety field generated by SVM.

SVM will learn the data to find the hyperplane which provide maximum margin to the two classes of data sets [25]. The detail of applying SVM presented in our previous work [18].

SVM will provide a safety field along the separating boundary as shown in Fig. 9. The gray amount represents the clearance of the cell to the obstacles. The pixel closer to the separating boundary will be safer.

The distance from the path point to the obstacles $d(p)$ is now calculated based on the distance to the hyperplane with

$$d(p) = \frac{|w \cdot p + b|}{||w||} \tag{8}$$

$w$ and $b$ are hyperplane margin and bias achieved from the training process.

Another benefit of this SVM applying step is that we can limit the searching area in the safety zone, around the separating boundary, so that the search space is reduced.

C. A continuous search space for parking path planning

Originally, the search space is portrayed as a 2D grid with each cell associates with a continuous 3D state $(x, y, \theta)$ of the vehicle. When a node is picked from the Front list of FMM, it is expanded by applying several steering actions (in our implementation, there are different steering angles from max-left to max-right) to the state associated with the node, and new children states are generated using a kinematic model of the vehicle:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} sin\theta \\ cos\theta \\ (tan\phi)/l \\ 0 \end{bmatrix} v_x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_y \tag{9}$$

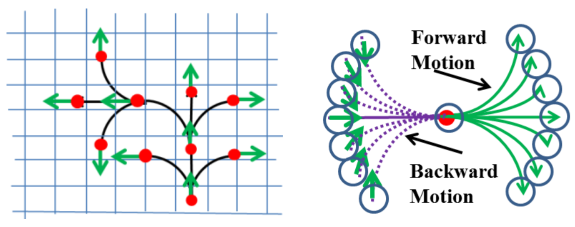


**Fig. 10** Search space.

where (x, y) represent the position of the center point of vehicle's rear wheel axis; $\theta$ is the vehicle's heading angle, $\phi$ is the steering angle, $v_x$ and $v_y$ are the longitudinal and rotational velocity and $l$ is the distance between the front and rear wheel axes.

For each of these continuous children states, we compute a grid cell that it falls into. Figure 10.a shows an example of the searching space graph based on 3 steering angles: max-right, 0 and max-left.

The cost value for travel between the nodes in the state space $f_{i.j}$ is calculated based on Eq. (2).

Different to the 2D case, the neighbors of each cell are based on the kinematic motion of the vehicle with considering different moving directions and steering angles. Figure 10.b shows the neighbors of one cell when apply 7 different steering angles for moving forward and backward.

D. Apply the FMM for path planning

Normally, the FMM will stop after the whole set of available nodes in the graph is examined. Therefore, to make the FMM algorithm more practically applicable, one condition will be added to make the algorithm stop when it reaches the goal position or the allowable runtime is reached.

```
3) Marching Loop
• Let p = (i_min, j_min) be the Front point with the smallest cost U;
    • If p is goal point then Goal_Reached = true
    • If Goal_Reached and max_run_time are reached then output current optimal path.
```

To make the algorithm convergence quicker to the goal point, the cost value 2D-dist-to-goal will be added when choosing the next point with smallest priority cost to be processed from the Front list.

$$Priority(p) = \mathrm{u}(p) + \lambda * 2D\text{-}dist\text{-}to\text{-}goal(p) \qquad (10)$$

Figure 11 shows an example of the planning results with the affection of $\lambda$ to the search space and the result path. When $\lambda$ is large (Fig. 11 b), the method will quickly advance to the goal point but the ending angle constraint will cause it to expend more search space around the goal point. When $\lambda$ value is small, the method is slowly exploring the map (Fig. 11 d). As shown in Fig. 11, the result paths are continuous, smooth and stay away from the obstacles. Moreover, the 2D-dist-to-goal can help reduce the search space therefore increase the speed of the algorithm.

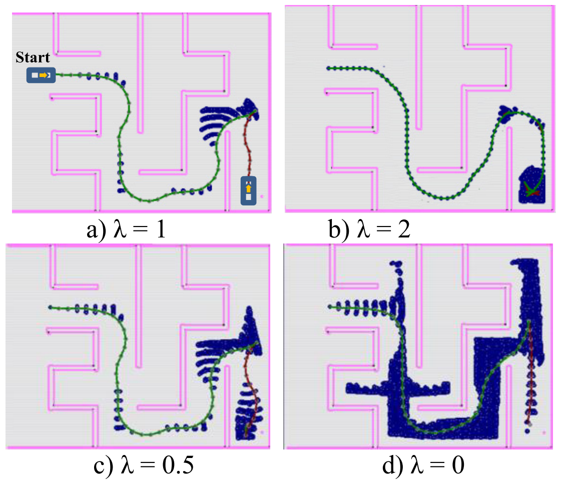


**Fig. 11** Global path generated by FMM.

After the global path is generated, the local path planner will be called to drive the vehicle follow this path and avoid new or moving obstacles.

## 4. Simulation and Experiment

The algorithms implemented in this section used the same cost function (2). The parameters we used are $w_1 = 2$, $w_2 = 5$, $w_3 = 1$, $w_4 = 5$ and $\lambda = 1$. We give more weight for gear transition, safety and smoothness. The travelled distance ($w_3$) and remain distance have equal weight.

### 4.1 Heuristic Evaluation

To evaluate the effectiveness of the 2D-dist-to-goal in Sect. 3.2 we compare our method with Gabriel's landmark-based for heuristically driven fast marching method [26]. In both methods, the nodes in the search graph are generated based on the kinematic model equation with the same constant velocity value and steering angle values. Gabriel method is applied with 40 landmarks uniformly distributed on the map boundary.

We tested both methods with random round shape obstacles (Case 1), a maze map with different local traps (Case 2 and 3). As shown in Fig. 12 and Table 1, our method is more effective for the maze environment or the map with local trap (e.g. U shape, V shape obstacles. . . ).

### 4.2 Path Planning for Parking

We implement our method and compare with the Dolgov-like-method [27]. The Dolgov-like method uses Hybrid-State A* (H-A*) associated with Risk Potential Map (RP). H-A* is a typical path planning method used by the Stanford Racing Team for the DARPA Challenge 2007. We implement the algorithms in different types of map and measure the average safety margin, curvature and gear transition of each results path corresponding to each case.

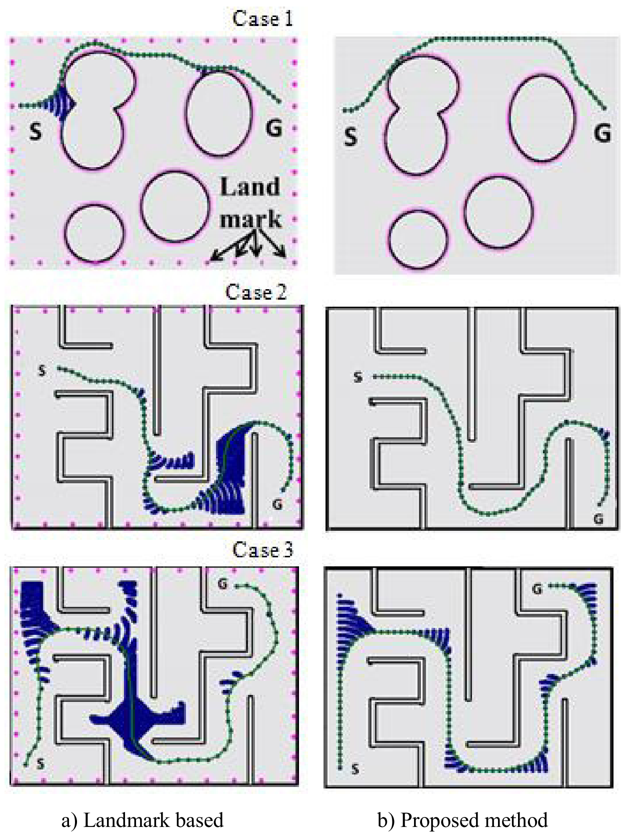


**Fig. 12** Search space evaluation.

**Table 1** Number of nodes generated.

| | Land-mark based | Proposed Method |
|---|---|---|
| Case 1 | 531 | 479 |
| Case 2 | 3071 | 1062 |
| Case 3 | 55099 | 4328 |

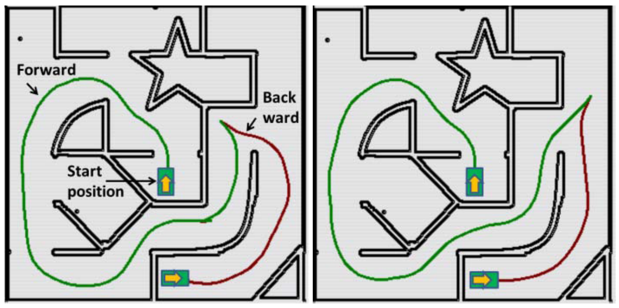


**Fig. 13** Case A: Simulation results of Hybrid A* (left) and proposed method (right) in unstructured map with different obstacles shapes.

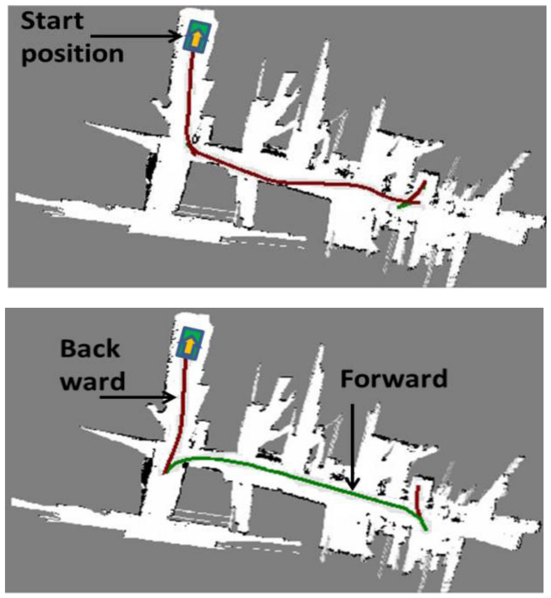


**Fig. 14** Case B: Simulation results of Hybrid A* (upper) and proposed method (lower) in narrow passage map with different obstacles shapes.

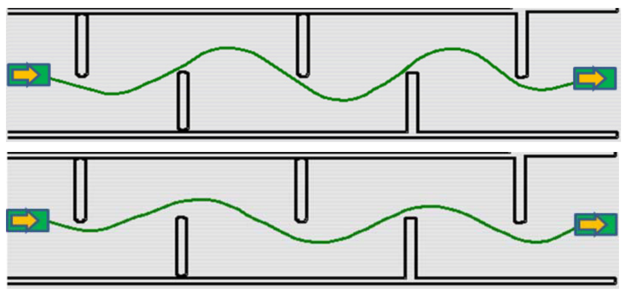


**Fig. 15** Case C: Simulation results of Hybrid A* (upper) and proposed method (lower) in ladder shape map.

### 4.2.1 Unstructured Map

We create simulation map to verify the algorithm in the conditions of different obstacles' shapes. As shown in Fig. 13, the obstacles have random shapes; the algorithm can generate a path that satisfies the vehicle dynamic. Figure 14 shows the SLAM map of a narrow passage environment. Figure 15 shows the paths generated in zig-zag type (or ladder shape) map.

### 4.2.2 Parkinglot Map

We apply the proposed global path planning algorithm in parking lot maps as shown in Fig. 16, Fig. 17 and Fig. 18. Figure 16 shows paths generated by both algorithms in structured parking lot map. Figure 17 shows paths generated in a SLAM map of a practical environment with other vehicles occupied parking lot. Figure 18 shows example paths generated by proposed method in the condition of other car (obstacle) show up on the lane to the parking position.

Figures 14, 16 and 18 are real environments recorded by onboard sensors to make a reference map. In order to verify proposed method in more complicated environment,

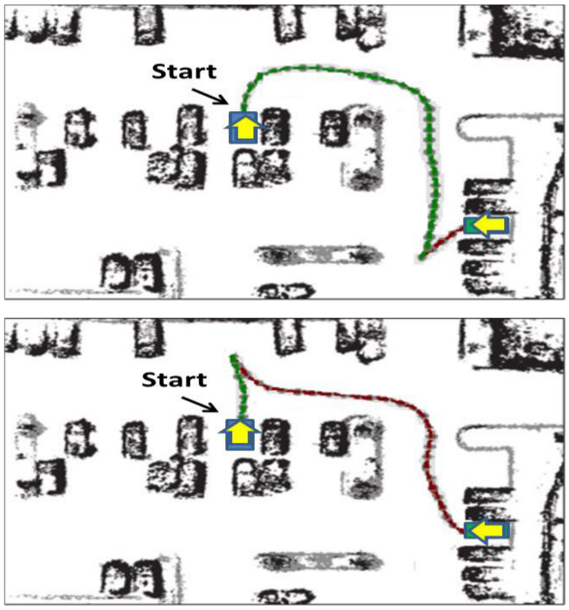


**Fig. 16** Case D Hybrid A* (upper) and proposed method (lower) generated path in a parking area SLAM map with the appearance of other cars.

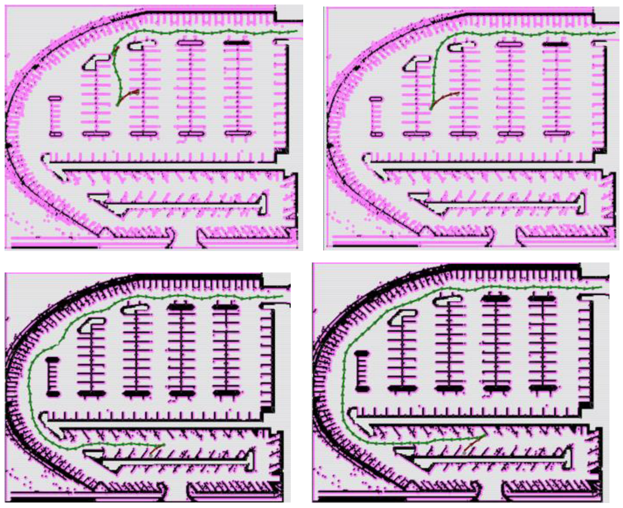


**Fig. 17** D Hybrid A* (left) and proposed method (right) generated path in a structured parking lot map in Case E (upper) and Case F (lower).

the map in Fig. 17 is artificially made.

Tables 2, 3, 4 and 5 correspondingly show a safety margin, a number of gear transition, average curvature of obtained paths and average computation time of proposed method and Hybrid A* method. Both algorithms are implemented in C++ on a 2.54Ghz Duo core computer.

The tested maps show various typical situation that may happen in path planning. In the situations which include narrow passage like case A, B, E our method generated paths with smaller number of gear change. Moreover, the average curvature is smaller and obstacle clearance is larger in every cases. Our proposed method provides a better quality path and SVM is faster than RP but for the whole algorithm, it needs more computation time. The global path

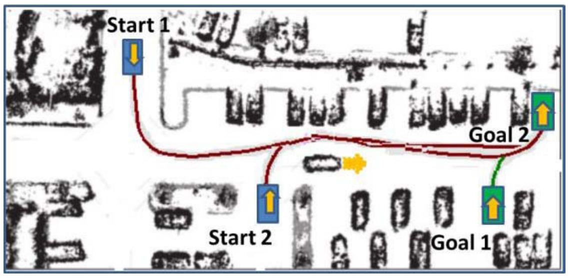


**Fig. 18** Path generated to park and avoid other moving vehicle interfere.

**Table 2** Safety Margin for Obtained Paths (cell).

| Method | H-A*+RP | FMM+SVM |
|---|---|---|
| Case A | 4.783 | 5.9611 |
| Case B | 17.3708 | 19.1896 |
| Case C | 4.0526 | 5.9344 |
| Case D | 7.3846 | 8.727 |
| Case E | 8.72 | 9,34 |
| Case F | 11.671 | 12.2631 |

**Table 3** Number of Gear change.

| Method | H-A*+RP | FMM+SVM |
|---|---|---|
| Case A | 3 | 1 |
| Case B | 4 | 2 |
| Case C | 1 | 1 |
| Case D | 1 | 1 |
| Case E | 3 | 1 |
| Case F | 1 | 1 |

**Table 4** Average curvature.

| Method | H-A*+RP | FMM+SVM |
|---|---|---|
| Case A | 0.243396 | 0.205583 |
| Case B | 0.218145 | 0.17069 |
| Case C | 0.1894 | 0.1045 |
| Case D | 0.29359 | 0.26818 |
| Case E | 0.25452 | 0.178 |
| Case F | 0.18815 | 0.19513 |

**Table 5** Average computation time (ms).

| Method | H-A*+RP | FMM+SVM |
|---|---|---|
| Case A | 234 | 561 |
| Case B | 145 | 317 |
| Case C | 59 | 229 |
| Case D | 116 | 343 |
| Case E | 271 | 612 |
| Case F | 308 | 836 |

planning is often used as a pre-calculated step before the vehicle travel and we call the local path planner after that to deal with dynamic obstacles so that this computation time could be acceptable.

## 5. Conclusion

We have presented a robust and practical method for autonomous vehicle parking path planning. The main contributions in our global path planning method are: (1) proposed an optimal path planning algorithm which considers the forward and backward moving maneuver, (2) the start and goal postures, the kinematic constraints, (3) the maximum clearance from path to obstacles. This method is based on FMM to find a parking path, SVM to obtain the obstacle clearance and dynamic vehicle model to generate a feasible path that satisfies the vehicle's constraints. The benefits of this approach in comparison with other typical related methods in the literature (including typical path planning method Potential Field [28], Probabilistic Roadmap [29], Rapidly Exploring Random Tree [30]) are: (1) this method works well with probabilistic SLAM map, (2) this method has no local minima problem, (3) the complexity of the method is not affected by the shape of the obstacles. Moreover, it can provide the most critical points within a path mathematically clearly and the generated path is smooth, feasible for the vehicle to follow. We also proposed schemes for autonomous parking which generated different behavior to deal with different situation while parking. The proposed method for automatic parking is effective to handle different parking situation include narrow passage cases.

### References

[1] C. Pradalier, S. Vaussier, and P. Corke, Path planning for a parking assistance system: Implementation and experimentation, Australian Robotics and Automation Association, Sydney, Australia, 2005.

[2] I.E. Paromtchik and C. Laugier, "Motion generation and control for parking an autonomous vehicle," Proc. IEEE International Conference on Robotics and Automation, pp.3117–3122, 1996.

[3] I.E. Paromtchik, "Steering and velocity commands for parking assistance," Proc. 10th IASTED International Conference on Robotics and Applications, pp.178–183, 2004.

[4] M.B. Oetiker, G.P. Baker, and L. Guzzella, "A navigation–field based semi–autonomous non–holonomic vehicle–parking assistant," IEEE Trans. Veh. Technol., vol.58, pp.1106–1118, 2009.

[5] P.E. Taix, M. Murray, R.M. Laumond, and J.-P. Jacobs, "A motion planner for nonholonomic mobile robots," IEEE Trans. Robot. Autom., vol.10, no.5, pp.577–593, 1994.

[6] J.D.B. Muller and S. Grodde, "Trajectory generation and feedforward control for parking a car," Proc. International Conference on Control Applications Munich, Germany, 2006.

[7] M.F. Hsieh and U. Ozguner, "A parking algorithm for an autonomous vehicle," Proc. IEEE Intelligent Vehicles Symposium, pp.1155–1160, 2008.

[8] K. Kondak and G. Hommel, "Computation of time optimal movements for autonomous parking of non-holonomic mobile platforms," International Conference on Robotics and Automation, pp.2698–2703, 2001.

[9] M. Wada, K.S. Yoon, and H. Hashimoto, "Development of advanced parking assistance system," IEEE Trans. Ind. Electron., vol.50, pp.4–17, 2003.

[10] B. Muller, J. Deutscher, and S. Grodde, "Continuous curvature trajectory design and feedforward control for parking a car," IEEE Trans. Control Syst. Technol., vol.15, pp.541–553, 2007.

[11] L.E. Dubins, "On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents," Amer. J. Math, pp.79:497–516, 1957.

[12] J.A. Reeds and R.A. Shepp, "Optimal paths for a car that goes both forward and backward," Pacific Journal of Mathematics, vol.145, no.2, pp.367–393, 1990.

[13] D. Ferguson, N. Kalra, and A. Stentz, "Replanning with RRTS," Proc. IEEE International Conference on Robotics and Automation, pp.1243–1248, 2006.

[14] L. Han, Q.H. Do, and S. Mita, "Unified path planner for parking an autonomous vehicle based on RRT," Proc. IEEE International Conference on Robotics and Automation, pp.5622–5627, 2011.

[15] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J.P. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," IEEE Trans. Control Syst. Technol., vol.17, no.5, pp.1105–1118, 2009.

[16] M. Likhachev and D. Ferguson, "Planning long dynamically-feasible maneuvers for autonomous vehicles," Int. J. Robotics Research (IJRR), vol.28, pp.933–945, 2009.

[17] Q.H. Do, L. Han, H.T.N. Nejad, and S. Mita, "Safe path planning among multi obstacles," IEEE Proc. Intelligent Vehicle Symposium, pp.332–338, 2011.

[18] Q.H. Do, S. Mita, H.T.N. Nejad, and L. Han, "Dynamic and safe path planning based on support vector machine among multi moving obstacles for autonomous vehicles," IEICE Trans. Inf. & Syst., vol.E96-D, no.2, pp.314–328, Feb. 2013.

[19] D. Adalsteinsson and J.A. Sethian, "A Fast level set method for propagating interfaces," J. Computational Physics, vol.118, pp.269–277, 1995.

[20] J.A. Sethian, "A marching level set method for monotonically advancing fronts," Proc. Nat. Acad. Sci., vol.93, no.4, pp.1591–1595, 1996.

[21] P. Melchior, B. Orsoni, O. Lavialle, A. Poty, and A. Oustaloup, "Consideration of obstacle danger level in path planning using A* and fast-marching optimization: comparative study," Signal Process., vol.11, pp.2387–2396, 2003.

[22] C.H. Chiang and P.J. Chiang, "A comparative study of implementing Fast Marching Method and A* SEARCH for mobile robot path planning in grid environment: Effect of map resolution," IEEE Proc. Advanced Robotics and Its Social Impacts, pp.1–6, 2007.

[23] S. Garrido, L. Moreno, and D. Blanco, "Voronoi diagram and fast marching applied to path planning," Proc. IEEE International Conference on Robotics and Automation, pp.3049–3054, 2006.

[24] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, and D. Lane, "Path planning for autonomous underwater vehicles," IEEE Trans. Robotics, vol.23, no.2, pp.331–341, April 2007.

[25] N. Cristianini and J.S. Taylor, An introduction to Support Vector Machines and other kernel-based learning methods (book style), Cambridge University Press, 2000.

[26] G. Peyre and La. Cohen, "Landmark-based geodesic computation for heuristically driven path planning," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp.2229–2236, June 2006.

[27] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," Proc. First International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR-08), June 2008.

[28] J. Barraquand, B. Langlois, and J.C. Latombe, "Numerical potential field techniques for robot path planning," IEEE Trans. Syst. Man Cybern., vol.22, no.2, pp.1012–1017, March 1992.

[29] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," IEEE Trans. Robot. Autom., vol.12, no.4, pp.566–580,

1996.
[30] S.M. Lavalle, "Rapidly random tree, a new tool for path planning," Technical report, Computer Science Dept., Iowa State University, Nov. 1998.

**Quoc Huy Do** received his B.S. and M.S. degrees in Information Technology from Hanoi University of Technology, Vietnam, in 2006 and 2008, respectively. He served on the faculty of Hanoi University of Science and Technology in the academic years of 2006–2009. He is currently a doctoral student at Toyota Technological Institute. His research field is path planning for autonomous vehicles.

**Seiichi Mita** received his B.S., M.S., and Ph.D. degrees in electrical engineering from Kyoto University in 1969, 1971, and 1989, respectively. He studied at Hitachi Central Research Laboratory, Kokubunji, Japan, from 1971 to 1991, investigating signal processing and coding methods. Currently, he is a professor at Toyota Technological Institute (TTI) in Nagoya (since 1999) and a director of the Research Center for Smart Vehicles at TTI. Currently, he is greatly interested in the research area of autonomous vehicles and sensing systems. Dr. Mita is a member of the Institute of Electronics, Information and Communication Engineers and the Institute of Image Information and Television Engineers in Japan. He is also a member of IEEE. He received the best paper award from the IEEE Consumer Electronics Society in 1986 and the best paper and author awards from the Institute of Television Engineers in Japan in 1987 and 1992, respectively.

**Keisuke Yoneda** received his B.S. degrees in engineering from Toyohashi University of Technology in 2007, and M.S., and Ph.D. degrees in information science from Hokkaido University in 2009 and 2012, respectively. He is currently a postdoctoral fellow at Toyota Technological Institute. He is interested in the research area of autonomous vehicles, artificial intelligence and artificial life. He is a member of the Japan Society for Precision Engineering and the Society of Automotive Engineers of Japan, Inc.